

[illegible]

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SSSSSS
LL          III          SSSSSS
LL          III          SS
LL          III          SS
LL          III          SS
LL          III          SS
LLLLLLLLLLLL IIIIIIII SSSSSSSS
LLLLLLLLLLLL IIIIIIII SSSSSSSS

```

```
1 0001 0 MODULE AED$SETACL (
2 0002 0 LANGUAGE (BLISS32),
3 0003 0 IDENT = 'V04-000',
4 0004 0 ADDRESSING_MODE (EXTERNAL = GENERAL)
5 0005 0 ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *****
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: SET utility
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 This module contains all the routines necessary to support the
38 0038 1 DCL commands SET FILE/ACL, SET DIRECTORY/ACL, SET DEVICE/ACL,
39 0039 1 and SET ACL with the exception of the /EDIT qualifier.
40 0040 1
41 0041 1 ENVIRONMENT:
42 0042 1
43 0043 1 VAX/VMS operating system, user mode utilities.
44 0044 1
45 0045 1 --
46 0046 1
47 0047 1
48 0048 1 AUTHOR: L. Mark Pilant CREATION DATE: 4-May-1983 9:20
49 0049 1
50 0050 1 MODIFIED BY:
51 0051 1
52 0052 1 V03-019 LMP0296 L. Mark Pilant, 6-Aug-1984 15:02
53 0053 1 Change the location of the code that determines if the target
54 0054 1 file is a directory file to correct a bug where the default
55 0055 1 option was being cleared.
56 0056 1
57 0057 1 V03-018 LMP0283 L. Mark Pilant, 25-Jul-1984 12:40
```


58	0058	1	Make sure the default object type is a file.
59	0059	1	
60	0060	1	V03-017 LMP0260 L. Mark Pilant, 27-Jun-1984 9:11
61	0061	1	Add support for the /DEFAULT qualifier.
62	0062	1	
63	0063	1	V03-016 LMP0253 L. Mark Pilant, 4-Jun-1984 10:41
64	0064	1	Fix the error handling in COPY_ACL so that SSS_NOMOREACE
65	0065	1	and SSS_ACLEMPY are (again) turned into SSS_NORMAL.
66	0066	1	
67	0067	1	V03-015 LMP0244 L. Mark Pilant, 1-May-1984 16:02
68	0068	1	Fix a bug intruded by LMP0238 that caused the wrong
69	0069	1	item code to be used.
70	0070	1	
71	0071	1	V03-014 LMP0238 L. Mark Pilant, 19-Apr-1984 13:35
72	0072	1	Use the size of the ACE being twiddled, when possible.
73	0073	1	
74	0074	1	V03-013 LMP0236 L. Mark Pilant, 18-Apr-1984 13:25
75	0075	1	Correct a bug that caused an ACCVIO to be returned from the
76	0076	1	\$CHANGE_ACL system service when an attempt was made to lock
77	0077	1	a file's ACL for writing.
78	0078	1	
79	0079	1	V03-012 LMP0230 L. Mark Pilant, 16-Apr-1984 10:45
80	0080	1	Track interface changes to \$CHANGE_ACL system service.
81	0081	1	
82	0082	1	V03-011 LMP0226 L. Mark Pilant, 9-Apr-1984 9:32
83	0083	1	Make sure all ACEs to be modified exist and are in the
84	0084	1	correct order (if more than one).
85	0085	1	
86	0086	1	V03-010 LMP0224 L. Mark Pilant, 7-Apr-1984 13:50
87	0087	1	Use enhanced lib\$file_scan features for stickyness.
88	0088	1	
89	0089	1	V03-009 LMP0223 L. Mark Pilant, 6-Apr-1984 12:49
90	0090	1	Use the correct amount of storage for the \$CHANGE_ACL
91	0091	1	lock block.
92	0092	1	
93	0093	1	V03-008 LMP0213 L. Mark Pilant, 24-Mar-1984 12:23
94	0094	1	Add support for locking and unlocking the object's ACL.
95	0095	1	Also, modify it so that the DCL commands SET ACL and SHOW
96	0096	1	ACL call the same image.
97	0097	1	
98	0098	1	V03-007 LMP0210 L. Mark Pilant, 23-Mar-1984 14:33
99	0099	1	Change the /MODIFY qualifier to /REPLACE.
100	0100	1	
101	0101	1	V03-006 LMP0198 L. Mark Pilant, 28-Feb-1984 12:05
102	0102	1	Open the object specified by the /LIKE qualifier for
103	0103	1	shared read access.
104	0104	1	
105	0105	1	V03-005 LMP0185 L. Mark Pilant, 4-Feb-1984 12:15
106	0106	1	Add support for device ACLs.
107	0107	1	
108	0108	1	V03-004 LMP0181 L. Mark Pilant, 15-Dec-1983 9:54
109	0109	1	Change code to use \$CHANGE_ACL instead of the ACP to do
110	0110	1	the ACL twiddling.
111	0111	1	
112	0112	1	V03-003 LMP0168 L. Mark Pilant, 11-Nov-1983 10:58
113	0113	1	Make use of the HIDDEN ACE option illegal.
114	0114	1	

AED\$SETACL
V04-000

I 13
16-Sep-1984 00:02:30 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:52:34 [ACLEDT.SRC]SETACL.B32;1

Page 3
(1)

```

: 115      0115 1 | V03-002 LMP0137      L. Mark Pilant,      12-Aug-1983  9:36
: 116      0116 1 |      Add support for the qualifiers: /BEFORE, /SINCE,
: 117      0117 1 |      and /CREATED.
: 118      0118 1 |
: 119      0119 1 | V03-001 LMP0126      L. Mark Pilant,      5-Jul-1983  11:00
: 120      0120 1 |      Correctly use a 'sticky' input file-spec. Also, handle
: 121      0121 1 |      errors while processing multiple files correctly.
: 122      0122 1 |
: 123      0123 1 | **
: 124      0124 1 |
: 125      0125 1 | LIBRARY 'SYSS$LIBRARY:LIB';
: 126      0126 1 | LIBRARY 'SYSS$LIBRARY:TPAMAC';
```

```
128 0127 1 ! Routines contained within this module.
129 0128 1
130 0129 1 FORWARD ROUTINE
131 0130 1 SET_ACL ! Main processing routine
132 0131 1 GET_FILE ! Get next output file spec
133 0132 1 PROCESS_FILE ! Act upon the specified file
134 0133 1 ADD_ACL ! Add to an existing ACL
135 0134 1 DELETE_ACL ! Delete ACEs or an ACL
136 0135 1 REPLACE_ACL ! Modify existing ACEs
137 0136 1 COPY_ACL ! Copy a object's ACL
138 0137 1 INPUT_ERROR ! Signal file scanning error
139 0138 1 FILE_ERROR ! Signal general file error
140 0139 1
141 0140 1 ! Define common error message codes.
142 0141 1
143 P 0142 1 $SHR_MSGDEF (SET, 119, LOCAL,
144 P 0143 1 (SYNTAX, SEVERE);
145 P 0144 1 (OPENIN, ERROR);
146 P 0145 1 (CLOSEIN, ERROR);
147 P 0146 1 (OPENOUT, ERROR);
148 P 0147 1 (CLOSEOUT, ERROR);
149 P 0148 1 (READERR, SEVERE);
150 P 0149 1 (WRITEERR, SEVERE);
151 0150 1 );
152 0151 1
153 0152 1 ! Define necessary macros.
154 0153 1
155 0154 1 MACRO
156 M 0155 1 SIGNAL (ARG) =
157 M 0156 1 BEGIN
158 M 0157 1 EXTERNAL ROUTINE LIB$SIGNAL;
159 M 0158 1 LIB$SIGNAL (ARG %IF %LENGTH-1 GTR 0 %THEN, %REMAINING %FI);
160 M 0159 1 IF NOT ARG AND
161 M 0160 1 (.WORST_ERROR AND ST$M_SEVERITY) LSS
162 M 0161 1 (ARG AND ST$M_SEVERITY) THEN WORST_ERROR = ARG OR
163 M 0162 1 ST$M_INHIB_MSG;
164 M 0163 1 END
165 0164 1 %;
166 0165 1
167 0166 1 MACRO
168 M 0167 1 ALLOCATE (SIZE, ADDRESS) =
169 M 0168 1 BEGIN
170 M 0169 1 EXTERNAL ROUTINE LIB$GET_VM;
171 M 0170 1 LOCAL VM_STATUS;
172 M 0171 1 VM_STATUS = LIB$GET_VM (%REF (SIZE), ADDRESS);
173 M 0172 1 IF .VM_STATUS THEN CH$FILL (0, SIZE, .ADDRESS);
174 M 0173 1 .VM_STATUS
175 M 0174 1 END
176 0175 1 %;
177 0176 1
178 0177 1 ! Various needed flags.
179 0178 1
180 0179 1 MACRO
181 0180 1 QUAL_AFTER = 0, 0, 1, 0 %; ! /AFTER qualifier seen
182 0181 1 QUAL_DELETE = 0, 1, 1, 0 %; ! /DELETE qualifier seen
183 0182 1 QUAL_LIKE = 0, 2, 1, 0 %; ! /LIKE qualifier seen
184 0183 1 QUAL_LOG = 0, 3, 1, 0 %; ! /LOG qualifier seen
```



```
185 0184 1 QUAL_REPLACE = 0, 4, 1, 0 %, ! /REPLACE qualifier seen
186 0185 1 QUAL_NEW = 0, 5, 1, 0 %, ! /NEW qualifier seen
187 0186 1 QUAL_DEFAULT = 0, 6, 1, 0 %, ! /DEFAULT qualifier seen
188 0187 1 DIRECTORY = 0, 10, 1, 0 %, ! Target file is a directory file
189 0188 1 IN_ELLIPSE = 0, 11, 1, 0 %, ! In ellipse processing
190 0189 1 SET_DEV_CMD = 0, 12, 1, 0 %, ! SET DEVICE command
191 0190 1 SET_FILE_CMD = 0, 13, 1, 0 %, ! SET FILE command
192 0191 1 SET_DIR_CMD = 0, 14, 1, 0 %, ! SET DIRECTORY command
193 0192 1 SET_ACL_CMD = 0, 15, 1, 0 %, ! SET ACL command
194 0193 1
195 0194 1 ! Structure definition for the old and new ACE queues.
196 0195 1
197 0196 1 MACRO
198 0197 1 ACEQ_L_FLINK = 0, 0, 32, 0 %, ! Forward link
199 0198 1 ACEQ_L_BLINK = 4, 0, 32, 0 %, ! Backward link
200 0199 1 ACEQ_T_ACE = 8, 0, 32, 0 %, ! Start of the actual ACE
201 0200 1
202 0201 1 LITERAL
203 0202 1 ACEQ_C_LENGTH = 8; ! Length of the overhead area
204 0203 1
205 0204 1 ! Semi-permanent storage.
206 0205 1
207 0206 1 OWN
208 0207 1 FLAGS : $BBLOCK [2], ! Needed flags
209 0208 1 WORST_ERROR, : ! Worst error encountered
210 0209 1 ACL_LOCKID : $BBLOCK [ACL$S_RLOCK_ACL], ! Lock-id for ACL lock
211 0210 1 OBJECT_TYPE, : ! Object type code
212 0211 1 OBJECT_NAME : $BBLOCK [DSC$C_S_BLN], ! Object name descriptor
213 0212 1 OBJECT_FAB : $FAB_DECL, ! Output object FAB
214 0213 1 OBJECT_NAME : $NAM_DECL, ! Output object NAME block
215 0214 1 OBJECT_EXP_NAME : $BBLOCK [NAM$C_MAXRSS], ! Expanded name string
216 0215 1 OBJECT_RES_NAME : $BBLOCK [NAM$C_MAXRSS], ! Resultant name string
217 0216 1 RELATED_NAME : $NAM_DECL, ! Related object spec
218 0217 1 CHAN, ! Input object channel
219 0218 1 ACL_CONTEXT, : $BBLOCK [ACL$S_RLOCK_ACL], ! ACL context used by $CHANGE_ACL
220 0219 1 SACL_LOCKID : ! Lock-id for ACL lock
221 0220 1 SUBJECT_TYPE, : $BBLOCK [DSC$C_S_BLN], ! Source object type code
222 0221 1 SUBJECT_DESC : ! Source object descr
223 0222 1 SUBJECT_FAB : $FAB_DECL, ! Source object FAB
224 0223 1 SUBJECT_NAME : $NAM_DECL, ! Source object NAME block
225 0224 1 SUBJECT_EXP_NAME : $BBLOCK [NAM$C_MAXRSS], ! Expanded name string
226 0225 1 SUBJECT_RES_NAME : $BBLOCK [NAM$C_MAXRSS], ! Resultant name string
227 0226 1 SCHAN, ! Source object channel
228 0227 1 SACL_CONTEXT, ! ACL context for $CHANGE_ACL
229 0228 1 SDEVICE_DESC : $BBLOCK [DSC$C_S_BLN], ! Source device desc
230 0229 1 SFIB_DESC : $BBLOCK [DSC$C_S_BLN], ! Source file FIB desc
231 0230 1 SFILE_FIB : $BBLOCK [FIB$C_LENGTH], ! Source file FIB
232 0231 1 COMMON_CTX, ! Common qual context
233 0232 1 ATR_ARGLIST : BLOCKVECTOR [3, ITM$S_ITEM, BYTE], ! ACP attribute descr
234 0233 1 CLI_ACE_DESC : $BBLOCK [DSC$C_S_BLN], ! ACE string from CLI
235 0234 1 ERROR_POS, ! Error position parsing ACE
236 0235 1 ACE_DESC : $BBLOCK [DSC$C_S_BLN], ! Binary ACE descriptor
237 0236 1 ACE : $BBLOCK [ACL$S_READACL], ! Binary ACE storage
238 0237 1 ACE_POINTER : REF $BBLOCK, ! Pointer to ACE queue entry
239 0238 1 ACE_TEXT_DESC : $BBLOCK [DSC$C_S_BLN], ! Text ACE descriptor
240 0239 1 ACE_TEXT : $BBLOCK [3072], ! AE text storage
241 0240 1 OLD_ACE_HEAD : $BBLOCK [ACEQ_C_LENGTH], ! Old ACE queue head
```

```
.. 242 0241 1 NEW_ACE HEAD : $BLOCK [ACEQ_C_LENGTH], ! New ACE queue head
.. 243 0242 1 DIR_GROUP, ! Group of UIC format directory
.. 244 0243 1 DIR_MEMBER; ! Member of UIC format directory
.. 245 0244 1
.. 246 0245 1 EXTERNAL
.. 247 0246 1 SET$NOHIDDEN, ! No HIDDEN ACEs allowed
.. 248 0247 1 SET$OBJLOCKED, ! Object locked by another user
.. 249 0248 1 SET$IVORDER, ! Incorrect ordering of ACEs to be modified
.. 250 0249 1 SET$NOSUCHACE, ! Specified ACE doesn't exist
.. 251 0250 1 SET$MODIFIED; ! Object modified message
.. 252 0251 1
.. 253 0252 1 EXTERNAL ROUTINE
.. 254 0253 1 CLISGET VALUE, ! Get qualifier value
.. 255 0254 1 CLISPRESENT, ! See if qualifier present
.. 256 0255 1 LIB$FID TO NAME, ! Translate FID to file-spec
.. 257 0256 1 LIB$FILE_SCAN, ! Search wildcard file spec
.. 258 0257 1 LIB$QUAL_FILE_MATCH, ! Check for match
.. 259 0258 1 LIB$QUAL_FILE_PARSE, ! Set match context
.. 260 0259 1 LIB$TPARSE; ! General purpose parser
.. 261 0260 1
.. 262 0261 1 ! TPARSE table for UIC format directory names.
.. 263 0262 1
.. 264 0263 1 $INIT_STATE (DIR_STATE, DIR_KEYS);
.. 265 0264 1
.. 266 0265 1 $STATE (,(TPAS_OCTAL,...,DIR_GROUP));
.. 267 0266 1 $STATE (,(','));
.. 268 0267 1 $STATE (,(TPAS_OCTAL,...,DIR_MEMBER));
```



```
270 0268 1 GLOBAL ROUTINE SET_ACL =
271 0269 1
272 0270 1 ++
273 0271 1
274 0272 1 FUNCTIONAL DESCRIPTION:
275 0273 1
276 0274 1 This routine is the main routine. It parses the command line to
277 0275 1 determine what modifications to the object (or objects) ACL are to
278 0276 1 occur.
279 0277 1
280 0278 1 --
281 0279 1
282 0280 2 BEGIN
283 0281 2
284 0282 2 BUILTIN
285 0283 2 INSQUE;
286 0284 2
287 0285 2 LOCAL
288 0286 2 SCAN_CONTEXT, ! LIB$FILE_SCAN context storage
289 0287 2 CMD_DESC : $BLOCK [DSC$S_BLN], ! DCL command descr
290 0288 2 STATUS, ! Local routine return status
291 0289 2 IO_STATUS : VECTOR [4, WORD]; ! I/O status block
292 0290 2
293 0291 2 ! Initialize local storage.
294 0292 2
295 0293 2 CH$FILL (0, 3*ITM$S_ITEM, ATR_ARGLIST);
296 0294 2 CH$FILL (0, FIB$C_LENGTH, SF$FILE_FIB);
297 0295 2 CH$FILL (0, DSC$S_BLN, CLI_ACE_DESC);
298 0296 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, ACE_DESC);
299 0297 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, ACE_TEXT_DESC);
300 0298 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, OBJECT_NAME);
301 0299 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, SUBJECT_DESC);
302 0300 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, CMD_DESC);
303 0301 2 CH$MOVE (DSC$S_BLN, CLI_ACE_DESC, SFIB_DESC);
304 0302 2
305 0303 2 FLAGS = 0;
306 0304 2 SCAN_CONTEXT = 0;
307 0305 2 OBJECT_TYPE = SUBJECT_TYPE = 0;
308 0306 2 CHAN = SCHAN = 0;
309 0307 2 WORST_ERROR = SS$NORMAL;
310 0308 2 CLI_ACE_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
311 0309 2 OBJECT_NAME[DSC$B_CLASS] = DSC$K_CLASS_D;
312 0310 2 SUBJECT_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
313 0311 2 CMD_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
314 0312 2 SFIB_DESC[DSC$B_LENGTH] = 10;
315 0313 2 SFIB_DESC[DSC$B_POINTER] = SF$FILE_FIB;
316 0314 2 ACE_DESC[DSC$B_POINTER] = ACE;
317 0315 2 OLD_ACE_HEAD[ACEQ_L_FLINK] = OLD_ACE_HEAD[ACEQ_L_BLINK]
318 0316 2 = OLD_ACE_HEAD[ACEQ_L_FLINK]; ! Null queue
319 0317 2 NEW_ACE_HEAD[ACEQ_L_FLINK] = NEW_ACE_HEAD[ACEQ_L_BLINK]
320 0318 2 = NEW_ACE_HEAD[ACEQ_L_FLINK]; ! Null queue
321 0319 2
322 0320 2 ! Determine what DCL command was used to invoke this image. Also, set the
323 0321 2 ! appropriate default object type code.
324 0322 2
325 0323 2 CL$GET VALUE ($DESCRIPTOR ('OPTION'), CMD_DESC);
326 0324 2 IF CH$EQL (.CMD_DESC[DSC$B_LENGTH], .CMD_DESC[DSC$B_POINTER],
```

```
327      MINU (.CMD_DESC[DSC$W_LENGTH], %CHARCOUNT ('FILE')), UPLIT ('FILE'),
328      0)
329  THEN
330  BEGIN
331  FLAGS[SET_FILE_CMD] = 1;
332  OBJECT_TYPE = ACL$C_FILE;
333  SUBJECT_TYPE = ACL$C_FILE;
334  END;
335  IF CH$EQL (.CMD_DESC[DSC$W_LENGTH], .CMD_DESC[DSC$A_POINTER],
336  MINU (.CMD_DESC[DSC$W_LENGTH], %CHARCOUNT ('DIRECTORY')), UPLIT ('DIRECTORY'),
337  0)
338  THEN
339  BEGIN
340  FLAGS[SET_DIR_CMD] = 1;
341  OBJECT_TYPE = ACL$C_FILE;
342  SUBJECT_TYPE = ACL$C_FILE;
343  END;
344  IF CH$EQL (.CMD_DESC[DSC$W_LENGTH], .CMD_DESC[DSC$A_POINTER],
345  MINU (.CMD_DESC[DSC$W_LENGTH], %CHARCOUNT ('DEVICE')), UPLIT ('DEVICE'),
346  0)
347  THEN
348  BEGIN
349  FLAGS[SET_DEV_CMD] = 1;
350  OBJECT_TYPE = ACL$C_DEVICE;
351  SUBJECT_TYPE = ACL$C_DEVICE;
352  END;
353  IF CH$EQL (.CMD_DESC[DSC$W_LENGTH], .CMD_DESC[DSC$A_POINTER],
354  MINU (.CMD_DESC[DSC$W_LENGTH], %CHARCOUNT ('ACL')), UPLIT ('ACL'),
355  0)
356  THEN
357  BEGIN
358  FLAGS[SET_ACL_CMD] = 1;
359  OBJECT_TYPE = ACL$C_FILE;
360  SUBJECT_TYPE = ACL$C_FILE;
361  END;
362  ! Determine what qualifiers are present.
363  FLAGS[QUAL_AFTER] = CL$PRESENT ($DESCRIPTOR ('AFTER'));
364  FLAGS[QUAL_DEFAULT] = CL$PRESENT ($DESCRIPTOR ('DEFAULT'));
365  FLAGS[QUAL_DELETE] = CL$PRESENT ($DESCRIPTOR ('DELETE'));
366  FLAGS[QUAL_LOG] = CL$PRESENT ($DESCRIPTOR ('LOG'));
367  FLAGS[QUAL_REPLACE] = CL$PRESENT ($DESCRIPTOR ('REPLACE'));
368  FLAGS[QUAL_NEW] = CL$PRESENT ($DESCRIPTOR ('NEW'));
369  ! If the /LIKE qualifier is present, get the source object type and name. If it
370  ! is a file, access it for later use.
371  IF (FLAGS[QUAL_LIKE] = CL$PRESENT ($DESCRIPTOR ('LIKE')))
372  THEN
373  BEGIN
374  ! Determine the characteristics of the source object.
```



```
384 0382 3 IF .FLAGS[SET_ACL_CMD]
385 0383 3 THEN
386 0384 3 BEGIN
387 0385 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.FILE')) THEN SUBJECT_TYPE = ACL$C_FILE;
388 0386 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.DEVICE')) THEN SUBJECT_TYPE = ACL$C_DEVICE;
389 0387 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.QUEUE')) THEN SUBJECT_TYPE = ACL$C_JOBCTL_QUEUE;
390 0388 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.EVENT CLUSTER')) THEN SUBJECT_TYPE = ACL$C_COMMON EF;
391 0389 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.LOGICAL NAME TABLE')) THEN SUBJECT_TYPE = ACL$C_LOGIC;
392 0390 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.PROCESS')) THEN SUBJECT_TYPE = ACL$C_PROCESS;
393 0391 3 IF CLISPRESNT ($DESCRIPTOR ('LIKE.OBJECT_TYPE.GLOBAL SECTION')) THEN SUBJECT_TYPE = ACL$C_GLOBAL_SE;
394 0392 3 CLISGET_VALUE ($DESCRIPTOR ('LIKE.OBJECT_NAME'), SUBJECT_DESC);
395 0393 3 END
396 0394 3 ELSE CLISGET_VALUE ($DESCRIPTOR ('LIKE'), SUBJECT_DESC);
397 0395 3
398 0396 3 ! Attempt to obtain a read lock for the source object.
399 0397 3
400 0398 3 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_RLOCK_ACL;
401 0399 3 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$C_RLOCK_ACL;
402 0400 3 ATR_ARGLIST[0, ITMSL_BUFADR] = SACL_LOCKID;
403 0401 3 STATUS = $CHANGE_ACL (CHAN = .SCHAN,
404 0402 3 OBJTYP = SUBJECT_TYPE,
405 0403 3 OBJNAM = SUBJECT_DESC,
406 0404 3 ITMLST = ATR_ARGLIST);
407 0405 3
408 0406 3 IF NOT .STATUS
409 0407 3 THEN
410 0408 3 BEGIN
411 0409 3 IF .STATUS EQL SSS NOTQUEUED
412 0410 3 THEN SIGNAL (SETS_OBJLOCKED)
413 0411 3 ELSE SIGNAL (.STATUS);
414 0412 3 RETURN .WORST_ERROR;
415 0413 3 END;
416 0414 3
417 0415 3 ! Open the source object to get the ACL being copied; if it is a file.
418 0416 3 IF .SUBJECT_TYPE EQL ACL$C_FILE
419 0417 3 THEN
420 0418 3 BEGIN
421 0419 3 $FAB_INIT (FAB = SUBJECT_FAB,
422 0420 3 FAC = GET,
423 0421 3 FNA = .SUBJECT_DESC[DSCSA_POINTER],
424 0422 3 FNS = .SUBJECT_DESC[DSCSW_LENGTH],
425 0423 3 FOP = UFO,
426 0424 3 NAM = SUBJECT_NAM,
427 0425 3 SHR = <GET, UPI>);
428 0426 3 $NAM_INIT (NAM = SUBJECT_NAM,
429 0427 3 ESA = SUBJECT_EXP_NAME,
430 0428 3 ESS = NAM$C_MAXRSS,
431 0429 3 RSA = SUBJECT_RES_NAME,
432 0430 3 RSS = NAM$C_MAXRSS);
433 0431 3 IF NOT $OPEN (FAB = SUBJECT_FAB)
434 0432 3 THEN
435 0433 3 BEGIN
436 0434 3 FILE_ERROR (SETS_OPENIN, SUBJECT_FAB, .SUBJECT_FAB[FAB$S_STS],
437 0435 3 .SUBJECT_FAB[FAB$S_STV]);
438 0436 3 RETURN SETS_OPENIN OR STSM_INHIB_MSG;
439 0437 3 END;
440 0438 3 SCHAN = .SUBJECT_FAB[FAB$S_STV];
```



```
441      0439      END;
442      0440      END;
443      0441
444      0442      ! Determine the characteristics of the target object.
445      0443
446      0444      IF .FLAGS[SET_ACL_CMD]
447      0445      THEN
448      0446      BEGIN
449      0447      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.FILE')) THEN OBJECT_TYPE = ACL$C_FILE;
450      0448      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.DEVICE')) THEN OBJECT_TYPE = ACL$C_DEVICE;
451      0449      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.QUEUE')) THEN OBJECT_TYPE = ACL$C_JOBCTL_QUEUE;
452      0450      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.EVENT CLUSTER')) THEN OBJECT_TYPE = ACL$C_COMMON EF CLUSTER;
453      0451      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.LOGICAL NAME TABLE')) THEN OBJECT_TYPE = ACL$C_LOGICAL_NAME_TA
454      0452      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.PROCESS')) THEN OBJECT_TYPE = ACL$C_PROCESS;
455      0453      IF CL$PRESENT ($DESCRIPTOR ('OBJECT_TYPE.GLOBAL_SECTION')) THEN OBJECT_TYPE = ACL$C_GLOBAL_SECTION;
456      0454      END;
457      0455
458      0456      ! Now get any ACEs specified on the /ACL qualifier.
459      0457
460      0458      WHILE CL$GET_VALUE ($DESCRIPTOR ('ACL'), CLI_ACE_DESC)
461      0459      DO
462      0460      BEGIN
463      0461      ACE_DESC[DSC$W_LENGTH] = ACL$S_READACL;           ! Reset buffer size
464      0462      STATUS = $PARSE_ACL (ACLSTR = CLI_ACE_DESC,
465      0463      ACLENT = ACE_DESC,
466      0464      ERRPOS = ERROR_POS);
467      0465
468      0466      IF NOT .STATUS
469      0467      THEN
470      0468      BEGIN
471      0469      CLI_ACE_DESC[DSC$A_POINTER] = .CLI_ACE_DESC[DSC$A_POINTER] + .ERROR_POS;
472      0470      CLI_ACE_DESC[DSC$W_LENGTH] = .CLI_ACE_DESC[DSC$W_LENGTH] - .ERROR_POS;
473      0471      SIGNAL (SETS_SYNTAX, 1, CLI_ACE_DESC, .STATUS, 0);
474      0472      RETURN .WORST_ERROR;
475      0473      END;
476      0474      IF .ACE[ACESV_HIDDEN]
477      0475      THEN
478      0476      BEGIN
479      0477      SIGNAL (SETS_NOHIDDEN);
480      0478      RETURN .WORST_ERROR;
481      0479      END;
482      0480      STATUS = ALLOCATE (.ACE[ACESB_SIZE] + ACEQ_C_LENGTH, ACE_POINTER);
483      0481      IF NOT .STATUS
484      0482      THEN
485      0483      BEGIN
486      0484      SIGNAL (.STATUS);
487      0485      RETURN .WORST_ERROR;
488      0486      END;
489      0487      CH$MOVE (.ACE[ACESB_SIZE], ACE, ACE_POINTER[ACEQ_T_ACE]);
490      0488      INSQUE (.ACE_POINTER, (IF .FLAGS[QUAL_DELETE] OR .FLAGS[QUAL_REPLACE]
491      0489      THEN .OLD_ACE_HEAD[ACEQ_L_BLINK]
492      0490      ELSE .NEW_ACE_HEAD[ACEQ_L_BLINK]));
493      0491      END;
494      0492      ! Now get any ACEs specified on the /REPLACE or /AFTER qualifiers.
495      0493
496      0494      WHILE CL$GET_VALUE ((IF .FLAGS[QUAL_REPLACE]
497      0495      THEN $DESCRIPTOR ('REPLACE'))
```

```
498 0496 2 ELSE $DESCRIPTOR ('AFTER')), CLI_ACE_DESC)
499 0497 DO
500 0498 BEGIN
501 0499 ACE_DESC[DSC$W_LENGTH] = ACL$S_READACL; ! Reset buffer size
502 P 0500 STATUS = $PARSE_ACL (ACLSTR = CLI_ACE_DESC,
503 P 0501 ACLENT = ACE_DESC,
504 0502 ERRPOS = ERROR_POS);
505 0503 IF NOT .STATUS
506 0504 THEN
507 0505 BEGIN
508 0506 CLI_ACE_DESC[DSC$A_POINTER] = .CLI_ACE_DESC[DSC$A_POINTER] + .ERROR_POS;
509 0507 CLI_ACE_DESC[DSC$W_LENGTH] = .CLI_ACE_DESC[DSC$W_LENGTH] - .ERROR_POS;
510 0508 SIGNAL (SETS_SYNTAX, 1, CLI_ACE_DESC, .STATUS, 0);
511 0509 RETURN .WORST_ERROR;
512 0510 END;
513 0511 IF .ACE[ACESV_HIDDEN]
514 0512 THEN
515 0513 BEGIN
516 0514 SIGNAL (SETS_NOHIDDEN);
517 0515 RETURN .WORST_ERROR;
518 0516 END;
519 0517 STATUS = ALLOCATE (.ACE[ACESB_SIZE] + ACEQ_C_LENGTH, ACE_POINTER);
520 0518 IF NOT .STATUS
521 0519 THEN
522 0520 BEGIN
523 0521 SIGNAL (.STATUS);
524 0522 RETURN .WORST_ERROR;
525 0523 END;
526 0524 CH$MOVE (.ACE[ACESB_SIZE], ACE, ACE_POINTER[ACEQ_T_ACE]);
527 0525 INSQUE (.ACE_POINTER, (IF .FLAGS[QUAL_REPLACE]
528 0526 THEN .NEW_ACE_HEAD[ACEQ_L_BLINK]
529 0527 ELSE .OLD_ACE_HEAD[ACEQ_L_BLINK]));
530 0528 END;
531 0529 ! Check for syntax errors on the command.
532 0530 IF .OLD_ACE_HEAD[ACEQ_L_FLINK] EQ LA OLD_ACE_HEAD[ACEQ_L_FLINK]
533 0531 AND .NEW_ACE_HEAD[ACEQ_C_FLINK] EQ LA NEW_ACE_HEAD[ACEQ_C_FLINK]
534 0532 THEN
535 0533 BEGIN
536 0534 IF .FLAGS[QUAL_AFTER] OR .FLAGS[QUAL_REPLACE]
537 0535 OR (.FLAGS[QUAL_NEW] AND NOT .FLAGS[QUAL_LIKE])
538 0536 THEN
539 0537 BEGIN
540 0538 SIGNAL (SETS_SYNTAX, 1, $DESCRIPTOR ('command line'));
541 0539 RETURN .WORST_ERROR;
542 0540 END;
543 0541 END
544 0542 ELSE
545 0543 BEGIN
546 0544 IF .FLAGS[QUAL_LIKE]
547 0545 THEN
548 0546 BEGIN
549 0547 SIGNAL (SETS_SYNTAX, 1, $DESCRIPTOR ('command line'));
550 0548 RETURN .WORST_ERROR;
551 0549 END;
552 0550 END;
553 0551
554 0552
```

```
555 0553 2
556 0554 2
557 0555 2 ! If the object is a file, loop through all the specifications supplied.
558 0556 2 ! For any other object, simply dispatch to the appropriate routine from here.
559 0557 2 IF .OBJECT_TYPE EQL ACL$C_FILE
560 0558 2 THEN
561 0559 2 BEGIN
562 0560 2     $FAB_INIT (FAB = OBJECT_FAB,
563 0561 2                 FAC = <GET, PUT>,
564 0562 2                 FOP = UFO,
565 0563 2                 NAM = OBJECT_NAM,
566 0564 2                 SHR = <GET, OPI>);
567 0565 2     $NAM_INIT (NAM = OBJECT_NAM,
568 0566 2                 ESA = OBJECT_EXP_NAME,
569 0567 2                 ESS = NAM$C_MAXRSS,
570 0568 2                 RSA = OBJECT_RES_NAME,
571 0569 2                 RSS = NAM$C_MAXRSS);
572 0570 2
573 0571 2 ! LIB$QUAL_FILE_PARSE is called to parse the common qualifiers. It sets up
574 0572 2 ! a data base which describes the results for LIB$QUAL_FILE_MATCH to use.
575 0573 2
576 0574 2     STATUS = LIB$QUAL_FILE_PARSE (%REF (LIB$M_CQF_BEFORE OR
577 0575 2                                     LIB$M_CQF_BYOWNER OR
578 0576 2                                     LIB$M_CQF_CONFIRM OR
579 0577 2                                     LIB$M_CQF_CREATED OR
580 0578 2                                     LIB$M_CQF_EXCLUDE OR
581 0579 2                                     LIB$M_CQF_SINCE), COMMON_CTX);
582 0580 2
583 0581 2 IF NOT .STATUS
584 0582 2 THEN
585 0583 2     BEGIN
586 0584 2         SIGNAL (.STATUS);
587 0585 2         RETURN .WORST_ERROR;
588 0586 2     END;
589 0587 2 ! Sit in a loop processing each 'input' file specified. For the copy
590 0588 2 ! operation, the 'input' file is really the output file.
591 0589 2
592 0590 2     FLAGS[IN ELLIPSE] = 0; ! for initial directory processing
593 0591 2     WHILE GET_FILE (OBJECT_FAB)
594 0592 2     DO
595 0593 2         BEGIN
596 0594 2
597 0595 2 ! If this is the /DEFAULT processing, and a channel has been assigned,
598 0596 2 ! deaccess the directory file, and deassign the channel.
599 0597 2
600 0598 2         IF .FLAGS[QUAL_DEFAULT] AND .SCHAN NEQ 0
601 0599 2         THEN
602 0600 2             BEGIN
603 0601 2                 STATUS = $QIOW (CHAN = .SCHAN,
604 0602 2                                 FUNC = IO$ DEACCESS,
605 0603 2                                 IOSB = IO_STATUS);
606 0604 2                 IF .STATUS THEN STATUS = .IO STATUS[0];
607 0605 2                 IF NOT .STATUS THEN SIGNAL (SET$ CLOSEIN, 1, SUBJECT_DESC, .STATUS, 0);
608 0606 2                 STATUS = $DASSGN (CHAN = .SCHAN);
609 0607 2                 IF NOT .STATUS THEN SIGNAL (SET$ CLOSEIN, 1, SUBJECT_DESC, .STATUS, 0);
610 0608 2
611 0609 2 ! Now release the read lock that was taken out for the directory file.
```



```

612 0610
613 0611 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_UNLOCK_ACL;
614 0612 ATR_ARGLIST[0, ITMSW_BUFSIZ] = 4;
615 0613 ATR_ARGLIST[0, ITMSL_BUFADR] = SACL_LOCKID;
616 0614 STATUS = $CHANGE_ACL (CHAN = .SCHAN,
617 0615 OBJTYP = OBJECT_TYPE,
618 0616 OBJNAM = OBJECT_DESC,
619 0617 ITMLST = ATR_ARGLIST);
620 0618 IF NOT .STATUS THEN SIGNAL (SET$C_CLOSEIN, 1, OBJECT_DESC, .STATUS, 0);
621 0619 SCHAN = 0;
622 0620 END;
623 0621 LIB$FILE_SCAN (OBJECT_FAB,
624 0622 PROCESS_FILE, ! File found action routine
625 0623 INPUT_ERROR, ! Input error action routine
626 0624 SCAN_CONTEXT); ! Stickiness context
627 0625 END;
628 0626 END
629 0627 ELSE
630 0628 BEGIN
631 0629 ! Get the object's name.
632 0630 CLISGET_VALUE ($DESCRIPTOR ('INPUT'), OBJECT_NAME);
633 0631
634 0632 ! Attempt to obtain a write lock for the target object.
635 0633
636 0634 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_WLOCK_ACL;
637 0635 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_WLOCK_ACL;
638 0636 ATR_ARGLIST[0, ITMSL_BUFADR] = ACL_LOCKID;
639 0637 STATUS = $CHANGE_ACL (CHAN = .CHAN,
640 0638 OBJTYP = OBJECT_TYPE,
641 0639 OBJNAM = OBJECT_NAME,
642 0640 ITMLST = ATR_ARGLIST);
643 0641
644 0642 IF NOT .STATUS
645 0643 THEN
646 0644 BEGIN
647 0645 IF .STATUS EQL SS$NOTQUEUED
648 0646 THEN SIGNAL (SET$OBJLOCKED)
649 0647 ELSE SIGNAL (.STATUS);
650 0648 RETURN .WORST_ERROR;
651 0649 END;
652 0650
653 0651 ! Call the necessary routine based upon the command line qualifiers.
654 0652
655 0653 IF .FLAGS[QUAL LIKE] THEN STATUS = COPY_ACL (OBJECT_NAME) ! /LIKE
656 0654 ELSE IF .FLAGS[QUAL DELETE] THEN STATUS = DELETE_ACL (OBJECT_NAME) ! /DELETE
657 0655 ELSE IF .FLAGS[QUAL REPLACE] THEN STATUS = REPLACE_ACL (OBJECT_NAME) ! /REPLACE
658 0656 ELSE STATUS = ADD_ACL (OBJECT_NAME); ! /AFTER, /NEW, or just /ACL
659 0657
660 0658 ! If logging is being done, indicate that the object has been modified.
661 0659
662 0660 IF .FLAGS[QUAL LOG] AND .STATUS
663 0661 THEN SIGNAL (SET$MODIFIED, 1, OBJECT_NAME);
664 0662 END;
665 0663
666 0664 RETURN .WORST_ERROR;
667 0665
668 0666

```

: 669

0667 1 END;

! End of routine SET_ACL

.TITLE AED\$SETACL
.IDENT \V04-000\

.PSECT _LIB\$STATES,NOWRT, SHR, PIC,1

00000 DIR_STATE::

.BLKB 0

45F4 00000 :TPASTYPE

U.2: .WORD 17908

00000000* 00002 :TPASADDR

U.3: .LONG <<DIR_GROUP-U.3>-4>

042C 00006 :TPASTYPE

U.4: .WORD 1068

45F4 00008 :TPASTYPE

U.5: .WORD 17908

00000000* 0000A :TPASADDR

U.6: .LONG <<DIR_MEMBER-U.6>-4>

.PSECT _LIB\$KEY0\$,NOWRT, SHR, PIC,1

00000 DIR_KEYS::

.BLKB 0

00000 :TPASKEY0

U.1: .BLKB 0

.PSECT \$SPLITS,NOWRT,NOEXE,2

4E 4F 49 54 50 4F 00000 P.AAB: .ASCII \OPTION\

00006 .BLKB 2

00000006 00008 P.AAA: .LONG 6

00000000* 0000C .ADDRESS P.AAB

00 00 00 59 52 4F 54 43 45 4C 49 46 00010 P.AAC: .ASCII \FILE\

00 00 45 43 49 56 45 44 00014 P.AAD: .ASCII \DIRECTORY\<0><0><0>

00 00 45 43 49 56 45 44 00020 P.AAE: .ASCII \DEVICE\<0><0>

00 00 45 43 49 56 45 44 00028 P.AAF: .ASCII \ACL\<0>

00 00 45 43 49 56 45 44 0002C P.AAH: .ASCII \AFTER\

00 00 45 43 49 56 45 44 00031 .BLKB 3

00 00 45 43 49 56 45 44 00034 P.AAG: .LONG 5

00 00 45 43 49 56 45 44 00038 .ADDRESS P.AAH

00 00 45 43 49 56 45 44 0003C P.AAJ: .ASCII \DEFAULT\

00 00 45 43 49 56 45 44 00043 .BLKB 1

00 00 45 43 49 56 45 44 00044 P.AAI: .LONG 7

00 00 45 43 49 56 45 44 00048 .ADDRESS P.AAJ

00 00 45 43 49 56 45 44 0004C P.AAL: .ASCII \DELETE\

00 00 45 43 49 56 45 44 00052 .BLKB 2

00 00 45 43 49 56 45 44 00054 P.AAK: .LONG 6

00 00 45 43 49 56 45 44 00058 .ADDRESS P.AAL

00 00 45 43 49 56 45 44 0005C P.AAN: .ASCII \LOG\

00 00 45 43 49 56 45 44 0005F .BLKB 1

00 00 45 43 49 56 45 44 00060 P.AAM: .LONG 3

00 00 45 43 49 56 45 44 00064 .ADDRESS P.AAN

00 00 45 43 49 56 45 44 00068 P.AAP: .ASCII \REPLACE\

00 00 45 43 49 56 45 44 0006F .BLKB 1

00 00 45 43 49 56 45 44 00070 P.AAO: .LONG 7

Page 15
(3)


```
00000012 001DA .BLKB 2
00000000 001DC P.ABO: .LONG 18
00000000 001E0 .ADDRESS P.ABP
45 55 51 2E 45 50 59 54 5F 54 43 45 4A 42 4F 001E4 P.ABR: .ASCII \OBJECT_TYPE.QUEUE\
45 55 001F3
00000011 001F5 .BLKB 3
00000000 001F8 P.ABQ: .LONG 17
00000000 001FC .ADDRESS P.ABR
45 56 45 2E 45 50 59 54 5F 54 43 45 4A 42 4F 00200 P.ABT: .ASCII \OBJECT_TYPE.EVENT_CLUSTER\
52 45 54 53 55 4C 43 5F 54 4E 0020F
00000019 00219 .BLKB 3
00000000 0021C P.ABS: .LONG 25
00000000 00220 .ADDRESS P.ABT
47 4F 4C 2E 45 50 59 54 5F 54 43 45 4A 42 4F 00224 P.ABV: .ASCII \OBJECT_TYPE.LOGICAL_NAME_TABLE\
45 4C 42 41 54 5F 45 4D 41 4E 5F 4C 41 43 49 00233
0000001E 00242 .BLKB 2
00000000 00244 P.ABU: .LONG 30
00000000 00248 .ADDRESS P.ABV
4F 52 50 2E 45 50 59 54 5F 54 43 45 4A 42 4F 0024C P.ABX: .ASCII \OBJECT_TYPE.PROCESS\
53 53 45 43 0025B
00000013 0025F .BLKB 1
00000000 00260 P.ABW: .LONG 19
00000000 00264 .ADDRESS P.ABX
4F 4C 47 2E 45 50 59 54 5F 54 43 45 4A 42 4F 00268 P.ABZ: .ASCII \OBJECT_TYPE.GLOBAL_SECTION\
4E 4F 49 54 43 45 53 5F 4C 41 42 00277
0000001A 00282 .BLKB 2
00000000 00284 P.ABY: .LONG 26
00000000 00288 .ADDRESS P.ABZ
4C 43 41 0028C P.ACB: .ASCII \ACL\
0028F .BLKB 1
00000003 00290 P.ACA: .LONG 3
00000000 00294 .ADDRESS P.ACB
45 43 41 4C 50 45 52 00298 P.ACD: .ASCII \REPLACE\
0029F .BLKB 1
00000007 002A0 P.ACC: .LONG 7
00000000 002A4 .ADDRESS P.ACD
52 45 54 46 41 002A8 P.ACF: .ASCII \AFTER\
002AD .BLKB 3
00000005 002B0 P.ACE: .LONG 5
00000000 002B4 .ADDRESS P.ACF
65 6E 69 6C 20 64 6E 61 6D 6D 6F 63 002B8 P.ACH: .ASCII \command line\
0000000C 002C4 P.ACG: .LONG 12
00000000 002C8 .ADDRESS P.ACH
65 6E 69 6C 20 64 6E 61 6D 6D 6F 63 002CC P.ACJ: .ASCII \command line\
0000000C 002D8 P.ACI: .LONG 12
00000000 002DC .ADDRESS P.ACJ
54 55 50 4E 49 002E0 P.ACL: .ASCII \INPUT\
002E5 .BLKB 3
00000005 002E8 P.ACK: .LONG 5
00000000 002EC .ADDRESS P.ACL

.PSECT $OWNS,NOEXE,2

00000 FLAGS: .BLKB 2
00002 .BLKB 2
00004 WORST_ERROR: .BLKB 4
```

J 14
16-Sep-1984 00:02:30
14-Sep-1984 11:52:34VAX-11 Bliss-32 V4.0-742
[ACLEDT.SRC]SETACL.B32;1Page 17
(3)

00008	ACL_LOCKID:	
	.BLKB	4
0000C	OBJECT_TYPE:	
	.BLKB	4
00010	OBJECT_NAME:	
	.BLKB	8
00018	OBJECT_FAB:	
	.BLKB	80
00068	OBJECT_NAM:	
	.BLKB	96
000C8	OBJECT_EXP NAME:	
	.BLKB	255
001C7		
	.BLKB	1
001C8	OBJECT_RES NAME:	
	.BLKB	255
002C7		
	.BLKB	1
002C8	RELATED_NAM:	
	.BLKB	96
00328	CHAN:	
	.BLKB	4
0032C	ACL_CONTEXT:	
	.BLKB	4
00330	SACL_LOCKID:	
	.BLKB	4
00334	SOBJECT_TYPE:	
	.BLKB	4
00338	SOBJECT_DESC:	
	.BLKB	8
00340	SOBJECT_FAB:	
	.BLKB	80
00390	SOBJECT_NAM:	
	.BLKB	96
003F0	SOBJECT_EXP NAME:	
	.BLKB	255
004EF		
	.BLKB	1
004F0	SOBJECT_RES NAME:	
	.BLKB	255
005EF		
	.BLKB	1
005F0	SCHAN:	
	.BLKB	4
005F4	SACL_CONTEXT:	
	.BLKB	4
005F8	SDEVICE_DESC:	
	.BLKB	8
00600	SFIB_DESC:	
	.BLKB	8
00608	SFILE_FIB:	
	.BLKB	64
00648	COMMON_CTX:	
	.BLKB	4
0064C	ATR_ARGLIST:	
	.BLKB	36
00670	CLI_ACE_DESC:	
	.BLKB	8
00678	ERROR_POS:	
	.BLKB	4
0067C	ACE_DESC:	
	.BLKB	8
00684	ACE:	
	.BLKB	512

00884 ACE_POINTER:
 .BKLB 4
00888 ACE_TEXT_DESC:
 .BKLB 8
00890 ACE_TEXT:
 .BKLB 3072
01490 OLD_ACE_HEAD:
 .BKLB 8
01498 NEW_ACE_HEAD:
 .BKLB 8
014A0 DIR_GROUP:
 .BKLB 4
014A4 DIR_MEMBER:
 .BKLB 4

\$RMS_PTR= SOBJECT_FAB
\$RMS_PTR= SOBJECT_NAM
\$RMS_PTR= OBJECT_FAB
\$RMS_PTR= OBJECT_NAM
 .EXTRN SET\$ _NOHIDDEN, SET\$ _OBJLOCKED
 .EXTRN SET\$ _IVORDER, SET\$ _ROSUCHACE
 .EXTRN SET\$ _MODIFIED, CLISGET VALUE
 .EXTRN CLISPRESENT, LIB\$FID TO NAME
 .EXTRN LIB\$FILE_SCAN, LIB\$QUAL_FILE_MATCH
 .EXTRN LIB\$QUAL_FILE_PARSE
 .EXTRN LIB\$TPARSE, SYSSCHANGE_ACL
 .EXTRN LIB\$SIGNAL, SYSSOPEN
 .EXTRN SYSSPARSE_ACL, LIB\$GET_VM
 .EXTRN SYSSQIOW, SYSSDASSGN

.PSECT \$CODE\$,NOWRT,2

OFFC 00000

.ENTRY SET_ACL, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,- 0268
 R11
MOVAB LIB\$SIGNAL, R11
MOVAB P.AAA, R10
MOVAB CLISPRESENT, R9
MOVAB FLAGS, R8
SUBL2 #24, SP
MOVCS #0, (SP), #0, #36, ATR_ARGLIST 0293
MOVCS #0, (SP), #0, #64, SFILE_FIB 0294
MOVCS #0, (SP), #0, #8, CLI_ACE_DESC 0295
MOVCS #8, CLI_ACE_DESC, ACE_DESC 0296
MOVCS #8, CLI_ACE_DESC, ACE_TEXT_DESC 0297
MOVCS #8, CLI_ACE_DESC, OBJECT_NAME 0298
MOVCS #8, CLI_ACE_DESC, SUBJECT_DESC 0299
MOVCS #8, CLI_ACE_DESC, CMD_DESC 0300
MOVCS #8, CLI_ACE_DESC, SFILE_DESC 0301
CLRW FLAGS 0303
CLRL SCAN_CONTEXT 0304
CLRL SUBJECT_TYPE 0305
CLRL OBJECT_TYPE
CLRL SCHAN
CLRL CHAN 0306

	24	00		5B	00000000G	00	9E	00002
				5A	0000	CF	9E	00009
				59	00000000G	00	9E	0000E
				58	0000	CF	9E	00015
				5E		18	C2	0001A
				6E		00	2C	0001D
					064C	C8		00022
0040	8F	00		6E		00	2C	00025
					0608	C8		0002C
	08	00		6E		00	2C	0002F
					0670	C8		00034
	067C	C8	0670	C8		08	28	00037
	0888	C8	0670	C8		08	28	0003F
	10	A8	0670	C8		08	28	00047
	0338	C8	0670	C8		08	28	0004E
	10	AE	0670	C8		08	28	00056
	0600	C8	0670	C8		08	28	0005D
						68	B4	00065
				04	AE	D4		00067
				0334	C8	D4		0006A
				0C	A8	D4		0006E
				05F0	C8	D4		00071
				0328	C8	D4		00075

		04	A8	01	D0	00079	MOVL	#1, WORST_ERROR	0307	
		0673	C8	02	90	0007D	MOVB	#2, CLI_ACE_DESC+3	0308	
		13	A8	02	90	00082	MOVB	#2, OBJECT_NAME+3	0309	
		0338	C8	02	90	00086	MOVB	#2, SUBJECT_DESC+3	0310	
		13	AE	02	90	0008B	MOVB	#2, CMD_DESC+3	0311	
		0600	C8	0A	B0	0008F	MOVW	#10, SFTB_DESC	0312	
		0604	C8	08	9E	00094	MOVAB	SFILE_FIB, SFTB_DESC+4	0313	
		0680	C8	08	9E	0009B	MOVAB	ACE, ACE_DESC+4	0314	
			50	1490	C8	9E	000A2	MOVAB	OLD_ACE_HEAD, R0	0316
		1494	C8	50	D0	000A7	MOVL	R0, OLD_ACE_HEAD+4		
		1490	C8	50	D0	000AC	MOVL	R0, OLD_ACE_HEAD		
			50	1498	C8	9E	000B1	MOVAB	NEW_ACE_HEAD, R0	0318
		149C	C8	50	D0	000B6	MOVL	R0, NEW_ACE_HEAD+4		
		1498	C8	50	D0	000BB	MOVL	R0, NEW_ACE_HEAD		
				10	AE	9F	000C0	PUSHAB	CMD_DESC	0323
					5A	DD	000C3	PUSHL	R10	
	000000006		00	02	FB	000C5	CALLS	#2, CLISGET_VALUE		
			54	10	AE	3C	000CC	MOVZWL	CMD_DESC, R4	0324
			50		54	D0	000D0	MOVL	R4, R0	0325
			04		50	B1	000D3	CMPW	R0, #4	
			50		03	1B	000D6	BLEQU	1\$	
50	00	14	50	04	D0	000D8	MOVL	#4, R0		
			BE	54	2D	000DB	CMPC5	R4, @CMD_DESC+4, #0, R0, P.AAC	0324	
				08	AA	000E1				
				0D	12	000E3	BNEQ	2\$		
		01	A8	20	88	000E5	BISB2	#32, FLAGS+1	0329	
		0C	A8	01	D0	000E9	MOVL	#1, OBJECT_TYPE	0330	
		0334	C8	01	D0	000ED	MOVL	#1, SUBJECT_TYPE	0331	
			50	54	D0	000F2	MOVL	R4, R0	0335	
			09	50	B1	000F5	CMPW	R0, #9		
			50	03	1B	000F8	BLEQU	3\$		
50	00	14	50	09	D0	000FA	MOVL	#9, R0		
			BE	54	2D	000FD	CMPC5	R4, @CMD_DESC+4, #0, R0, P.AAD	0334	
				0C	AA	00103				
				40	0E	12	00105	BNEQ	4\$	
		01	A8	8F	88	00107	BISB2	#64, FLAGS+1	0339	
		0C	A8	01	D0	0010C	MOVL	#1, OBJECT_TYPE	0340	
		0334	C8	01	D0	00110	MOVL	#1, SUBJECT_TYPE	0341	
			50	54	D0	00115	MOVL	R4, R0	0345	
			06	50	B1	00118	CMPW	R0, #6		
			50	03	1B	0011B	BLEQU	5\$		
50	00	14	50	06	D0	0011D	MOVL	#6, R0		
			BE	54	2D	00120	CMPC5	R4, @CMD_DESC+4, #0, R0, P.AAE	0344	
				18	AA	00126				
					0D	12	00128	BNEQ	6\$	
		01	A8	10	88	0012A	BISB2	#16, FLAGS+1	0349	
		0C	A8	02	D0	0012E	MOVL	#2, OBJECT_TYPE	0350	
		0334	C8	02	D0	00132	MOVL	#2, SUBJECT_TYPE	0351	
			50	54	D0	00137	MOVL	R4, R0	0355	
			03	50	B1	0013A	CMPW	R0, #3		
			50	03	1B	0013D	BLEQU	7\$		
50	00	14	50	03	D0	0013F	MOVL	#3, R0		
			BE	54	2D	00142	CMPC5	R4, @CMD_DESC+4, #0, R0, P.AAF	0354	
				20	AA	00148				
				80	0E	12	0014A	BNEQ	8\$	
		01	A8	8F	88	0014C	BISB2	#128, FLAGS+1	0359	
		0C	A8	01	D0	00151	MOVL	#1, OBJECT_TYPE	0360	

68	01	0334	C8	2C	01	DO	00155	8\$:	MOVL	#1, SUBJECT_TYPE	0361
			69		AA	9F	0015A		PUSHAB	P.AAG	0366
			00		01	FB	0015D		CALLS	#1, CLISPRESNT	
				3C	50	FO	00160		INSV	R0, #0, #1, FLAGS	
			69		AA	9F	00165		PUSHAB	P.AAI	0367
68	01		06		01	FB	00168		CALLS	#1, CLISPRESNT	
				4C	50	FO	0016B		INSV	R0, #6, #1, FLAGS	
68	01		01		AA	9F	00170		PUSHAB	P.AAK	0368
			69		01	FB	00173		CALLS	#1, CLISPRESNT	
68	01		01		50	FO	00176		INSV	R0, #1, #1, FLAGS	
			69	58	AA	9F	0017B		PUSHAB	P.AAM	0369
			03		01	FB	0017E		CALLS	#1, CLISPRESNT	
68	01				50	FO	00181		INSV	R0, #3, #1, FLAGS	
			69	68	AA	9F	00186		PUSHAB	P.AAO	0370
			04		01	FB	00189		CALLS	#1, CLISPRESNT	
68	01				50	FO	0018C		INSV	R0, #4, #1, FLAGS	
			69	74	AA	9F	00191		PUSHAB	P.AAQ	0371
			05		01	FB	00194		CALLS	#1, CLISPRESNT	
68	01				50	FO	00197		INSV	R0, #5, #1, FLAGS	
			69	0080	CA	9F	0019C		PUSHAB	P.AAS	0376
			02		01	FB	001A0		CALLS	#1, CLISPRESNT	
68	01		03		50	FO	001A3		INSV	R0, #2, #1, FLAGS	
					50	EB	001AB		BLBS	R0, 9\$	
				01	015A	31	001AB	9\$:	BRW	22\$	
					A8	95	001AE		TSTB	FLAGS+1	0382
					73	18	001B1		BGEQ	17\$	
			69	00A0	CA	9F	001B3		PUSHAB	P.AAU	0385
			05		01	FB	001B7		CALLS	#1, CLISPRESNT	
			0334	C8	50	E9	001BA		BLBC	R0, 10\$	
				00C0	01	DO	001BD	10\$:	MOVL	#1, SUBJECT_TYPE	0386
			69		CA	9F	001C2		PUSHAB	P.AAW	
			05		01	FB	001C6		CALLS	#1, CLISPRESNT	
0334	C8				50	E9	001C9		BLBC	R0, 11\$	
			69	00E0	02	DO	001CC	11\$:	MOVL	#2, SUBJECT_TYPE	0387
			05		CA	9F	001D1		PUSHAB	P.AAY	
			0334	C8	01	FB	001D5		CALLS	#1, CLISPRESNT	
					50	E9	001D8		BLBC	R0, 12\$	
			69	0108	03	DO	001DB	12\$:	MOVL	#3, SUBJECT_TYPE	0388
			05		CA	9F	001E0		PUSHAB	P.ABA	
0334	C8				01	FB	001E4		CALLS	#1, CLISPRESNT	
			69		50	E9	001E7		BLBC	R0, 13\$	
			0334	C8	04	DO	001EA	13\$:	MOVL	#4, SUBJECT_TYPE	0389
				0134	CA	9F	001EF		PUSHAB	P.ABC	
			69		01	FB	001F3		CALLS	#1, CLISPRESNT	
			05		50	E9	001F6		BLBC	R0, 14\$	
0334	C8				05	DO	001F9	14\$:	MOVL	#5, SUBJECT_TYPE	0390
			69	0154	CA	9F	001FE		PUSHAB	P.ABE	
			05		01	FB	00202		CALLS	#1, CLISPRESNT	
			0334	C8	50	E9	00205		BLBC	R0, 15\$	
			69	017C	06	DO	00208	15\$:	MOVL	#6, SUBJECT_TYPE	0391
			05		CA	9F	0020D		PUSHAB	P.ABG	
0334	C8				01	FB	00211		CALLS	#1, CLISPRESNT	
			69		50	E9	00214		BLBC	R0, 16\$	
			0334	C8	07	DO	00217	16\$:	MOVL	#7, SUBJECT_TYPE	0392
				0338	C8	9F	0021C		PUSHAB	SUBJECT_DESC	
				0194	CA	9F	00220		PUSHAB	P.ABI	
					08	11	00224		BRB	18\$	

			0338	C8	9F	00226	17%:	PUSHAB	SUBJECT_DESC		0394
			01A0	CA	9F	0022A		PUSHAB	P.ABK		
	00000000G	00		02	FB	0022E	18%:	CALLS	#2, CLISGET VALUE		0399
	064C	C8	000A0004	8F	D0	00235		MOVL	#655364, ATR_ARGLIST		0400
	0650	C8	0330	C8	9E	0023E		MOVAB	SACL_LOCKID, ATR_ARGLIST+4		0404
				7E	7C	00245		CLRD	-(SP)		
				7E	D4	00247		CLRL	-(SP)		
			064C	C8	9F	00249		PUSHAB	ATR_ARGLIST		
			0338	C8	9F	0024D		PUSHAB	SUBJECT_DESC		
			0334	C8	9F	00251		PUSHAB	SUBJECT_TYPE		
			05F0	C8	DD	00255		PUSHL	SCHAN		
	00000000G	00		07	FB	00259		CALLS	#7, SYSSCHANGE_ACL		
		57		50	D0	00260		MOVL	R0, STATUS		
		03		57	E8	00263		BLBS	STATUS, 19\$		0405
			0334	0489	31	00266		BRW	71\$		
		01		C8	D1	00269	19%:	CPL	SUBJECT_TYPE, #1		0416
				03	13	0026E		BEQL	20\$		
0050	8F	00		0095	31	00270		BRW	22\$		
				00	2C	00273	20%:	MOVC5	#0, (SP), #0, #80, \$RMS_PTR		0425
			0340	C8		0027A					
	0340	C8	5003	8F	B0	0027D		MOVW	#20483, \$RMS_PTR		
	0344	C8	00020000	8F	D0	00284		MOVL	#131072, \$RMS_PTR+4		
	0356	C8	4202	8F	B0	0028D		MOVW	#16898, \$RMS_PTR+22		
	035F	C8		02	90	00294		MOVB	#2, \$RMS_PTR+31		
	0368	C8	0390	C8	9E	00299		MOVAB	SUBJECT_NAME, \$RMS_PTR+40		
	036C	C8	033C	C8	D0	002A0		MOVL	SUBJECT_DESC+4, \$RMS_PTR+44		
0060	8F	00	0374	C8	90	002A7		MOVB	SUBJECT_DESC, \$RMS_PTR+52		
				00	2C	002AE		MOVC5	#0, (SP), #0, #96, \$RMS_PTR		0430
			0390	C8		002B5					
	0390	C8	6002	8F	B0	002B8		MOVW	#24578, \$RMS_PTR		
	0392	C8		01	8E	002BF		MNEGB	#1, \$RMS_PTR+2		
	0394	C8	04F0	C8	9E	002C4		MOVAB	SUBJECT_RES_NAME, \$RMS_PTR+4		
	039A	C8		01	8E	002CB		MNEGB	#1, \$RMS_PTR+10		
	039C	C8	03F0	C8	9E	002D0		MOVAB	SUBJECT_EXP_NAME, \$RMS_PTR+12		
			0340	C8	9F	002D7		PUSHAB	SUBJECT_FAB		0431
	00000000G	00		01	FB	002DB		CALLS	#1, SYSOPEN		
		1C		50	E8	002E2		BLBS	R0, 21\$		
		7E	0348	C8	7D	002E5		MOVQ	SUBJECT_FAB+8, -(SP)		0434
			0340	C8	9F	002EA		PUSHAB	SUBJECT_FAB		
			0077109A	8F	DD	002EE		PUSHL	#780303\$		
	0000V	CF		04	FB	002F4		CALLS	#4, FILE_ERROR		
		50	1077109A	8F	D0	002F9		MOVL	#276238490, R0		0436
				04		00300		RET			
	05F0	C8	034C	C8	D0	00301	21%:	MOVL	SUBJECT_FAB+12, SCHAN		0438
			01	A8	95	00308	22%:	TSTB	FLAGS+1		0444
				62	18	0030B		BGEQ	29\$		
			01B8	CA	9F	0030D		PUSHAB	P.ABM		0447
		69		01	FB	00311		CALLS	#1, CLISPRESENT		
	0C	A8		50	E9	00314		BLBC	R0, 23\$		
			01D4	01	D0	00317		MOVL	#1, OBJECT_TYPE		0448
		69		CA	9F	0031B	23%:	PUSHAB	P.ABO		
		04		01	FB	0031F		CALLS	#1, CLISPRESENT		
	0C	A8		50	E9	00322		BLBC	R0, 24\$		
			01F0	02	D0	00325		MOVL	#2, OBJECT_TYPE		
		69		CA	9F	00329	24%:	PUSHAB	P.ABQ		0449
		04		01	FB	0032D		CALLS	#1, CLISPRESENT		
				50	E9	00330		BLBC	R0, 25\$		

OC	A8	0214	03	DO	00333	MOV	#3, OBJECT_TYPE	0450
	69		CA	9F	00337	PUSHAB	P.ABS	
	04		01	FB	00338	CALLS	#1, CLISPRESENT	
OC	A8	023C	50	E9	0033E	BLBC	RO, 26\$	0451
	69		04	DO	00341	MOV	#4, OBJECT_TYPE	
	04		CA	9F	00345	PUSHAB	P.ABU	
	69		01	FB	00349	CALLS	#1, CLISPRESENT	
OC	A8	0258	50	E9	0034C	BLBC	RO, 27\$	0452
	69		05	DO	0034F	MOV	#5, OBJECT_TYPE	
	04		CA	9F	00353	PUSHAB	P.ABW	
	69		01	FB	00357	CALLS	#1, CLISPRESENT	
OC	A8	027C	50	E9	0035A	BLBC	RO, 28\$	0453
	69		06	DO	0035D	MOV	#6, OBJECT_TYPE	
	04		CA	9F	00361	PUSHAB	P.ABY	
	69		01	FB	00365	CALLS	#1, CLISPRESENT	
OC	A8	0670	50	E9	00368	BLBC	RO, 29\$	0458
	04		07	DO	0036B	MOV	#7, OBJECT_TYPE	
	69		C8	9F	0036F	PUSHAB	CLI_ACE_DESC	
00000000G	00	0288	CA	9F	00373	PUSHAB	P.ACA	
	03		02	FB	00377	CALLS	#2, CLISGET_VALUE	
			50	E9	0037E	BLBS	RO, 30\$	
067C	C8	0200	00DB	31	00381	BRW	41\$	0461
			8F	B0	00384	MOVW	#512, ACE_DESC	
			7E	D4	00388	CLRL	-(SP)	0464
		0678	C8	9F	0038D	PUSHAB	ERROR_POS	
		067C	C8	9F	00391	PUSHAB	ACE_DESC	
		0670	C8	9F	00395	PUSHAB	CLI_ACE_DESC	
00000000G	00		04	FB	00399	CALLS	#4, SYSPARSE_ACL	
	57		50	DO	003A0	MOV	RO, STATUS	
	24		57	E8	003A3	BLBS	STATUS, 32\$	0465
0674	C8	0678	C8	C0	003A6	ADDL2	ERROR_POS, CLI_ACE_DESC+4	0468
0670	C8	0678	C8	A2	003AD	SUBW2	ERROR_POS, CLI_ACE_DESC	0469
			7E	D4	003B4	CLRL	-(SP)	0470
			57	DD	003B6	PUSHL	STATUS	
		0670	C8	9F	003B8	PUSHAB	CLI_ACE_DESC	
			01	DD	003BC	PUSHL	#1	
		007710FC	8F	DD	003BE	PUSHL	#7803132	
	68		05	FB	003C4	CALLS	#5, LIB\$SIGNAL	
			00DD	31	003C7	BRW	45\$	
2D	0687	C8	02	E1	003CA	BBC	#2, ACE+3, 35\$	0473
		00000000G	00	9F	003D0	PUSHAB	SET\$ NOHIDDEN	0476
	68		01	FB	003D6	CALLS	#1, LIB\$SIGNAL	
	50	00000000G	00	9E	003D9	MOVAB	SET\$ NOHIDDEN, RO	
	17		50	E8	003E0	BLBS	RO, 34\$	
	50	00000000*	00	9E	003E3	MOVAB	<SET\$ NOHIDDEN?>, RO	
50	04	A8	03	ED	003EA	CMPZV	#0, #3, WORST_ERROR, RO	
			08	18	003F0	BGEQ	34\$	
	04	A8	00	9E	003F2	MOVAB	<SET\$ NOHIDDEN!268435456>, WORST_ERROR	0477
			03E5	31	003FA	BRW	79\$	0479
		0884	C8	9F	003FD	PUSHAB	ACE_POINTER	
	04	0684	C8	9A	00401	MOVZBL	ACE, 4(SP)	
	04		08	C0	00407	ADDL2	#8, 4(SP)	
		04	AE	9F	0040B	PUSHAB	4(SP)	
00000000G	00		02	FB	0040E	CALLS	#2, LIB\$GET_VM	
	56		50	DO	00415	MOV	RO, VM STATUS	
	10		56	E9	00418	BLBC	VM STATUS, 36\$	
	50	0684	C8	9A	0041B	MOVZBL	ACE, RO	

50	00	50	08	CO	00420	ADDL2	#8, R0		
		6E	00	2C	00423	MOVCS	#0, (SP), #0, R0, @ACE_POINTER		
			0884	D8	00428				
		57	56	DO	00428	36\$:	MOVL	VM STATUS, STATUS	
		03	57	E8	0042E		BLBS	STATUS, 37\$	0480
			0323	31	00431		BRW	72\$	
		50	0684	C8	9A	37\$:	MOVZBL	ACE, R0	0486
		56	0884	C8	DO		MOVL	ACE_POINTER, R6	
08	A6	0684	C8	50	28		MOVCS	R0, ACE, 8(R6)	
	04		68	01	E0		BBS	#1, FLAGS, 38\$	0487
	07		68	04	E1		BBC	#4, FLAGS, 39\$	
		50	1494	C8	DO	38\$:	MOVL	OLD_ACE_HEAD+4, R0	0488
				05	11		BRB	40\$	
		50	149C	C8	DO	39\$:	MOVL	NEW_ACE_HEAD+4, R0	0489
		60		66	0E	40\$:	INSQUE	(R6), (R0)	0487
			FF10	31	0045C		BRW	29\$	0458
			0670	C8	9F	41\$:	PUSHAB	CLI_ACE_DESC	0494
	07	68		04	E1		BBC	#4, FLAGS, 42\$	
		50	0298	CA	9E		MOVAB	P.ACC, R0	0495
				05	11		BRB	43\$	
		50	02A8	CA	9E	42\$:	MOVAB	P.ACE, R0	0496
				50	DD	43\$:	PUSHL	R0	
		00000000G	00	02	FB		CALLS	#2, CLISGET_VALUE	0494
			03	50	E8		BLBS	R0, 44\$	
			067C	C8	009A		BRW	53\$	
				8F	B0	44\$:	MOVW	#512, ACE_DESC	0499
				7E	D4		CLRL	-(SP)	0502
			0678	C8	9F		PUSHAB	ERROR_POS	
			067C	C8	9F		PUSHAB	ACE_DESC	
			0670	C8	9F		PUSHAB	CLI_ACE_DESC	
				04	FB		CALLS	#4, SYSPARSE_ACL	
		00000000G	00	50	DO		MOVL	R0, STATUS	
			57	57	E8		BLBS	STATUS, 47\$	0503
			11	FEFF	31		BRW	31\$	0506
04	04	A8	03	00	ED	45\$:	CMPZV	#0, #3, WORST_ERROR, #4	0508
				03	18		BGEQ	46\$	
				00AD	31		BRW	57\$	
				032D	31		BRW	79\$	
		03	0687	C8	02		BBC	#2, ACE+3, 48\$	0511
				FF12	31		BRW	33\$	
			0884	C8	9F	48\$:	PUSHAB	ACE_POINTER	0517
			0684	C8	9A		MOVZBL	ACE, 4(SP)	
		04	AE	08	CO		ADDL2	#8, 4(SP)	
		04	AE	9F	004CC		PUSHAB	4(SP)	
		00000000G	00	02	FB		CALLS	#2, LIB\$GET_VM	
			56	50	DO		MOVL	R0, VM STATUS	
			10	56	E9		BLBC	VM STATUS, 49\$	
			50	0684	C8		MOVZBL	ACE, R0	
			50	08	CO		ADDL2	#8, R0	
50	00	6E	0884	00	2C		MOVCS	#0, (SP), #0, R0, @ACE_POINTER	
				D8	004E9				
		57	56	DO	004EC	49\$:	MOVL	VM STATUS, STATUS	
		03	57	E8	004EF		BLBS	STATUS, 50\$	0518
			0262	31	004F2		BRW	72\$	
		50	0684	C8	9A	50\$:	MOVZBL	ACE, R0	0524
		56	0884	C8	DO		MOVL	ACE_POINTER, R6	
08	A6	0684	C8	50	28		MOVCS	R0, ACE, 8(R6)	

07	68	04	E1	00506	BBC	#4, FLAGS, 51%	0525			
	50	149C	C8	DO	0050A	MOVL	NEW_ACE_HEAD+4, R0	0526		
			05	11	0050F	BRB	52%			
	50	1494	C8	DO	00511	51%:	MOVL	OLD_ACE_HEAD+4, R0	0527	
	60		66	OE	00516	52%:	INSQUE	(R6), (R0)	0525	
			FF43	31	00519	BRW	41%		0494	
	50	1490	C8	9E	0051C	53%:	MOVAB	OLD_ACE_HEAD, R0	0532	
	50	1490	C8	D1	00521		CMPL	OLD_ACE_HEAD, R0		
			2D	12	00526		BNEQ	56%		
	50	1498	C8	9E	00528		MOVAB	NEW_ACE_HEAD, R0	0533	
	50	1498	C8	D1	0052D		CMPL	NEW_ACE_HEAD, R0		
			21	12	00532		BNEQ	56%		
	0C		68	E8	00534		BLBS	FLAGS, 54%	0536	
08	68		04	EO	00537		BBS	#4, FLAGS, 54%		
2B	68		05	E1	00538		BBC	#5, FLAGS, 58%	0537	
27	68		02	EO	0053F		BBS	#2, FLAGS, 58%		
		028C	CA	9F	00543	54%:	PUSHAB	P.ACG	0540	
			01	DD	00547	55%:	PUSHL	#1		
		007710FC	8F	DD	00549		PUSHL	#7803132		
	6B		03	FB	0054F		CALLS	#3, LIB\$SIGNAL		
			FF52	31	00552		BRW	45%		
11	68		02	E1	00555	56%:	BBC	#2, FLAGS, 58%	0546	
		02D0	CA	9F	00559		PUSHAB	P.ACI	0549	
			E8	11	0055D		BRB	55%		
	04	A8	107710FC	8F	DO	0055F	57%:	MOVL	#276238588, WORST_ERROR	
			0278	31	00567		BRW	79%	0550	
	01	0C	A8	D1	0056A	58%:	CMPL	OBJECT_TYPE, #1	0557	
			03	13	0056E		BEQL	59%		
			0173	31	00570		BRW	70%		
0050	BF	00	6E	00	2C	00573	59%:	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	0564
			18	A8		0057A				
	18	A8	5003	8F	BO	0057C		MOVW	#20483, \$RMS_PTR	
	1C	A8	00020000	8F	DO	00582		MOVL	#131072, \$RMS_PTR+4	
	2E	A8	4203	8F	BO	0058A		MOVW	#16899, \$RMS_PTR+22	
	37	A8		02	90	00590		MOVW	#2, \$RMS_PTR+31	
	40	A8	68	A8	9E	00594		MOVAB	OBJECT_NAME, \$RMS_PTR+40	
0060	BF	00	6E	00	2C	00599		MOVCS	#0, (SP), #0, #96, \$RMS_PTR	0569
			68	A8		005A0				
	68	A8	6002	8F	BO	005A2		MOVW	#24578, \$RMS_PTR	
	6A	A8		01	8E	005A8		MNEGB	#1, \$RMS_PTR+2	
	6C	A8	01C8	C8	9E	005AC		MOVAB	OBJECT_RES_NAME, \$RMS_PTR+4	
	72	A8		01	8E	005B2		MNEGB	#1, \$RMS_PTR+10	
	74	A8	00C8	C8	9E	005B6		MOVAB	OBJECT_EXP_NAME, \$RMS_PTR+12	
			0648	C8	9F	005BC		PUSHAB	COMMON_CTX	0574
	04	AE	011F	8F	3C	005C0		MOVZWL	#287, 4(SP)	0578
			04	AE	9F	005C6		PUSHAB	4(SP)	0574
	00000000G	00		02	FB	005C9		CALLS	#2, LIB\$QUAL_FILE_PARSE	
		57		50	DO	005D0		MOVL	R0, STATUS	
		03		57	E8	005D3		BLBS	STATUS, 60%	0580
			017E	31	005D6		BRW	72%		
	01	A8		08	8A	005D9	60%:	BICB2	#8, FLAGS+1	0590
			18	A8	9F	005DD	61%:	PUSHAB	OBJECT_FAB	0591
	0000V	CF		01	FB	005E0		CALLS	#1, GET_FILE	
		03		50	E8	005E5		BLBS	R0, 62%	
			01F7	31	005E8		BRW	79%		
03		68		06	EO	005EB	62%:	BBS	#6, FLAGS, 64%	0598
			00DC	31	005EF	63%:	BRW	69%		

			05F0	C8	D5	005F2	64\$:	TSTL	SCHAN		
				F7	13	005F6		BEQL	63\$		
				7E	7C	005F8		CLRQ	-(SP)		0603
				7E	7C	005FA		CLRQ	-(SP)		
				7E	7C	005FC		CLRQ	-(SP)		
				7E	7C	005FE		CLRQ	-(SP)		
			28	AE	9F	00600		PUSHAB	IO STATUS		
				34	DD	00603		PUSHL	#52		
			05F0	C8	DD	00605		PUSHL	SCHAN		
				7E	D4	00609		CLRL	-(SP)		
		00000000G	00	0C	FB	0060B		CALLS	#12, SYS\$QIOW		
			57	50	DD	00612		MOVL	RO, STATUS		0604
			07	57	E9	00615		BLBC	STATUS, 65\$		
			57	08	AE	3C	00618	MOVZWL	IO STATUS, STATUS		
			23	57	E8	0061C		BLBS	STATUS, 66\$		0605
				7E	D4	0061F	65\$:	CLRL	-(SP)		
				57	DD	00621		PUSHL	STATUS		
			0338	C8	9F	00623		PUSHAB	SUBJECT_DESC		
				01	DD	00627		PUSHL	#1		
			00771052	8F	DD	00629		PUSHL	#7802962		
				05	FB	0062F		CALLS	#5, LIB\$SIGNAL		
02	04	A8	6B	00	ED	00632		CMPZV	#0, #3, WORST_ERROR, #2		
			03	08	18	00638		BGEQ	66\$		
				08	18	00638		MOV	#276238418, WORST_ERROR		0606
		04	A8	8F	DD	0063A		MOV	#276238418, WORST_ERROR		
			05F0	C8	DD	00642	66\$:	PUSHL	SCHAN		
		00000000G	00	01	FB	00646		CALLS	#1, SYS\$DASSGN		
			57	50	DD	0064D		MOVL	RO, STATUS		
			23	57	E8	00650		BLBS	STATUS, 67\$		0607
				7E	D4	00653		CLRL	-(SP)		
				57	DD	00655		PUSHL	STATUS		
			0338	C8	9F	00657		PUSHAB	SUBJECT_DESC		
				01	DD	0065B		PUSHL	#1		
			00771052	8F	DD	0065D		PUSHL	#7802962		
				05	FB	00663		CALLS	#5, LIB\$SIGNAL		
02	04	A8	6B	00	ED	00666		CMPZV	#0, #3, WORST_ERROR, #2		
			03	08	18	0066C		BGEQ	67\$		
				08	18	0066C		MOV	#276238418, WORST_ERROR		0612
		04	A8	8F	DD	0066E		MOV	#786436, ATR_ARGLIST		0613
		064C	C8	8F	DD	00676	67\$:	MOVAB	SACL LOCKID, ATR_ARGLIST+4		
		0650	C8	000C0004	8F	DD		MOVAB	SACL LOCKID, ATR_ARGLIST+4		0617
			0330	C8	9E	0067F		CLRQ	-(SP)		
				7E	7C	00686		CLRL	-(SP)		
				7E	D4	00688		CLRL	-(SP)		
			064C	C8	9F	0068A		PUSHAB	ATR_ARGLIST		
			0338	C8	9F	0068E		PUSHAB	SUBJECT_DESC		
			0334	C8	9F	00692		PUSHAB	SUBJECT_TYPE		
			05F0	C8	DD	00696		PUSHL	SCHAN		
		00000000G	00	07	FB	0069A		CALLS	#7, SYS\$CHANGE_ACL		
			57	50	DD	006A1		MOVL	RO, STATUS		
			23	57	E8	006A4		BLBS	STATUS, 68\$		0618
				7E	D4	006A7		CLRL	-(SP)		
				57	DD	006A9		PUSHL	STATUS		
			0338	C8	9F	006AB		PUSHAB	SUBJECT_DESC		
				01	DD	006AF		PUSHL	#1		
			00771052	8F	DD	006B1		PUSHL	#7802962		
				05	FB	006B7		CALLS	#5, LIB\$SIGNAL		
02	04	A8	6B	00	ED	006BA		CMPZV	#0, #3, WORST_ERROR, #2		
			03	08	18	006C0		BGEQ	68\$		
				08	18	006C0		MOV	#276238418, WORST_ERROR		
		04	A8	8F	DD	006C2		MOV	#276238418, WORST_ERROR		

			05F0	C8	D4	006CA	688:	CLRL	SCHAN	0619
			04	AE	9F	006CE	698:	PUSHAB	SCAN CONTEXT	0621
			0000V	CF	9F	006D1		PUSHAB	INPUT_ERROR	
			0000V	CF	9F	006D5		PUSHAB	PROCESS_FILE	
			18	AB	9F	006D9		PUSHAB	OBJECT_FAB	
		00000000G	00	04	FB	006DC		CALLS	#4, LIB\$FILE_SCAN	
				FEF7	31	006E3		BRW	61\$	0591
			10	AB	9F	006E6	708:	PUSHAB	OBJECT_NAME	0632
			02E0	CA	9F	006E9		PUSHAB	P_ACK	
		00000000G	00	02	FB	006ED		CALLS	#2, CLISGET_VALUE	
		064C	C8	8F	D0	006F4		MOVL	#720900, ATR_ARGLIST	0637
		0650	C8	AB	9E	006FD		MOVAB	ACL_LOCKID, ATR_ARGLIST+4	0638
				7E	7C	00703		CLRG	-(SP)	0642
				7E	D4	00705		CLRL	-(SP)	
		064C		C8	9F	00707		PUSHAB	ATR_ARGLIST	
		10		AB	9F	0070B		PUSHAB	OBJECT_NAME	
		0C		AB	9F	0070E		PUSHAB	OBJECT_TYPE	
		0328		C8	DD	00711		PUSHL	CHAN	
		00000000G	00	07	FB	00715		CALLS	#7, SYS\$CHANGE_ACL	
		57		50	D0	0071C		MOVL	R0, STATUS	
		55		57	E8	0071F		BLBS	STATUS, 74\$	0643
		000009B8	8F	57	D1	00722	718:	CMPL	STATUS, #2488	0646
				2C	12	00729		BNEQ	72\$	
			00000000G	00	9F	0072B		PUSHAB	SET\$ OBJLOCKED	0647
		6B		01	FB	00731		CALLS	#1, CIB\$SIGNAL	
		50	00000000G	00	9E	00734		MOVAB	SET\$ OBJLOCKED, R0	
		37		50	E8	0073B		BLBS	R0, 73\$	
50	04	AB	50	00000000*	00	9E	0073E	MOVAB	<SET\$ OBJLOCKED&7>, R0	
			03	00	ED	00745		CMPZV	#0, #3, WORST_ERROR, R0	
				28	18	0074B		BGEQ	73\$	
		04	AB	00000000*	00	9E	0074D	MOVAB	<SET\$ OBJLOCKED!268435456>, WORST_ERROR	
				1E	11	00755		BRB	73\$	0646
				57	DD	00757	728:	PUSHL	STATUS	0648
		6B		01	FB	00759		CALLS	#1, LIB\$SIGNAL	
		16		57	E8	0075C		BLBS	STATUS, 73\$	
50		57	03	00	EF	0075F		EXTZV	#0, #3, STATUS, R0	
50	04	AB	03	00	ED	00764		CMPZV	#0, #3, WORST_ERROR, R0	
				76	18	0076A		BGEQ	79\$	
	04	AB	57	10000000	8F	C9	0076C	BISL3	#268435456, STATUS, WORST_ERROR	
				6B	11	00775	738:	BRB	79\$	0649
		0A		02	E1	00777	748:	BBC	#2, FLAGS, 75\$	0654
			10	AB	9F	0077B		PUSHAB	OBJECT_NAME	
		0000V	CF	01	FB	0077E		CALLS	#1, COPY_ACL	
				24	11	00783		BRB	78\$	
		0A		01	E1	00785	758:	BBC	#1, FLAGS, 76\$	0655
			10	AB	9F	00789		PUSHAB	OBJECT_NAME	
		0000V	CF	01	FB	0078C		CALLS	#1, DELETE_ACL	
				16	11	00791		BRB	78\$	
		0A		04	E1	00793	768:	BBC	#4, FLAGS, 77\$	0656
			10	AB	9F	00797		PUSHAB	OBJECT_NAME	
		0000V	CF	01	FB	0079A		CALLS	#1, REPLACE_ACL	
				08	11	0079F		BRB	78\$	
			10	AB	9F	007A1	778:	PUSHAB	OBJECT_NAME	0657
		0000V	CF	01	FB	007A4		CALLS	#1, ADD_ACL	
				50	D0	007A9	788:	MOVL	R0, STATUS	
32			57	03	E1	007AC		BBC	#3, FLAGS, 79\$	0661
			68	57	E9	007B0		BLBC	STATUS, 79\$	
			2F							

AED\$SETACL
V04-000

6 15
16-Sep-1984 00:02:30 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:52:34 [ACLEDT.SRC]SETACL.B32;1

Page 27
(3)

50	04	A8	10	A8	9F	007B3	PUSHAB	OBJECT_NAME	0662
				01	DD	007B6	PUSHL	#1	
			00000000G	00	9F	007B8	PUSHAB	SETS_MODIFIED	
68				03	FB	007BE	CALLS	#3, [IBSSIGNAL	
50			00000000G	00	9E	007C1	MOVAB	SETS_MODIFIED, R0	
17				50	EB	007C8	BLBS	R0, 798	
50			00000000*	00	9E	007CB	MOVAB	<SETS_MODIFIED&7>, R0	
03				00	ED	007D2	CMPZV	#0, #3, WORST_ERROR, R0	
				08	18	007D8	BGEQ	798	
04	A8	00000000*		00	9E	007DA	MOVAB	<SETS_MODIFIED!268435456>, WORST_ERROR	
50		04	A8	00	9E	007E2	MOVL	WORST_ERROR, R0	0665
				04	007E6	798:	RET		0667

: Routine Size: 2023 bytes, Routine Base: \$CODE\$ + 0000


```
0671 0668 1 ROUTINE GET_FILE =
0672 0669 1
0673 0670 1 ++
0674 0671 1
0675 0672 1 FUNCTIONAL DESCRIPTION:
0676 0673 1
0677 0674 1 This routine gets the next file specification in the command line.
0678 0675 1 If there are no more specifications, the routine returns zero.
0679 0676 1 Otherwise, the next file specification is placed in the specified
0680 0677 1 FAB for later searching and parsing.
0681 0678 1
0682 0679 1 CALLING SEQUENCE:
0683 0680 1
0684 0681 1 GET_FILE
0685 0682 1
0686 0683 1 INPUT PARAMETERS:
0687 0684 1 none
0688 0685 1
0689 0686 1 INPLICIT INPUTS:
0690 0687 1 none
0691 0688 1
0692 0689 1 OUTPUT PARAMETERS:
0693 0690 1 none
0694 0691 1
0695 0692 1 IMPLICIT OUTPUTS:
0696 0693 1 none
0697 0694 1
0698 0695 1 ROUTINE VALUE:
0699 0696 1 1 if a specification was found
0700 0697 1 0 otherwise
0701 0698 1
0702 0699 1 SIDE EFFECTS:
0703 0700 1 The retrieved file specification is placed into the specified FAB.
0704 0701 1
0705 0702 1 --
0706 0703 1
0707 0704 2 BEGIN
0708 0705 2
0709 0706 2 OWN
0710 0707 2 FILE_DESC : $BLOCK [DSC$C_S_BLN] ! File name descr
0711 0708 2 INITIAL (REP DSC$C_S_BLN OF (BYTE (0)));
0712 0709 2
0713 0710 2 LOCAL
0714 0711 2 DESC : $BLOCK [DSC$C_S_BLN], ! Temp descriptor
0715 0712 2 ENDCHAR : BYTE, ! Dir spec terminator
0716 0713 2 EOS, ! End addr of dir spec
0717 0714 2 PTR, ! Moving pointer in dir spec
0718 0715 2 STR_PTR, ! Pointer to remainder of spec
0719 0716 2 STR_LEN, ! Remaining length of dir spec
0720 0717 2 TEMP_STRING : VECTOR [NAM$C_MAXRSS, BYTE], ! Temp dir spec storage
0721 0718 2 TEMP, ! Location of string to find
0722 0719 2 STATUS: ! Local routine exit status
0723 0720 2
0724 0721 2 ! Determine whether or not it is necessary to get another input specification.
0725 0722 2
0726 0723 3 IF NOT .FLAGS[SET_DIR_CMD] OR (.FLAGS[SET_DIR_CMD] AND NOT .FLAGS[IN_ELLIPSE])
0727 0724 2 THEN
```

```
728 BEGIN
729
730 ! If there are no more specifications, return 0.
731
732 FILE_DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
733 IF NOT CLISGET_VALUE ($DESCRIPTOR ('INPUT'), FILE_DESC) THEN RETURN 0;
734
735 ! Fill in the FAB fields for the normal (or simple) case.
736
737 OBJECT_FAB[FAB$L_FNA] = .FILE_DESC[DSC$A_POINTER];
738 OBJECT_FAB[FAB$B_FNS] = .FILE_DESC[DSC$W_LENGTH];
739 END;
740
741 ! If this is a SET DIRECTORY command, it is necessary to do some additional
742 ! processing of the input file specification. In other words, it will be
743 ! necessary to turn the directory specification into a file specification.
744
745 IF .FLAGS[SET_DIR_CMD]
746 THEN
747 BEGIN
748
749 ! Check here to see if a trailing ellipse is being treated. If so,
750 ! then FLAGS[IN_ELLIPSE] will be set to 1, and there's no need
751 ! to search and see if such a trailing ellipse is present. However,
752 ! if the value is set to 0, then get a new directory spec.
753
754 IF NOT .FLAGS[IN_ELLIPSE] ! If not processing an ellipse
755 THEN ! then get the next directory
756 BEGIN
757 OBJECT_FAB[FAB$L_FNA] = .FILE_DESC[DSC$A_POINTER];
758 OBJECT_FAB[FAB$B_FNS] = .FILE_DESC[DSC$W_LENGTH];
759
760 ! Since this is a new entry, it must be checked for a trailing ellipse.
761
762 CH$MOVE (.FILE_DESC[DSC$W_LENGTH], ! Move this many chars
763 .FILE_DESC[DSC$A_POINTER], ! From the CLI area
764 TEMP_STRING); ! To the temp string
765 STR_PTR = TEMP_STRING; ! Set up pointer
766 STR_LEN = .FILE_DESC[DSC$W_LENGTH]; ! and length.
767
768 ! Look for ellipses.
769
770 WHILE NOT CH$FAIL (TEMP = CH$FIND_SUB (.STR_LEN, .STR_PTR,
771 3, UPLIT ('...')))
772 DO
773 BEGIN
774 STR_PTR = .TEMP + 3; ! Update pointer
775 STR_LEN = .STR_LEN - (.TEMP - .STR_PTR) - 3;
776 END;
777
778 ! After the final ellipse, check to see if it is at the end of the
779 ! directory specification. If so, then change the context field of
780 ! the fab, and insert an end bracket at the beginning of the ellipse.
781
782 IF (.STR_PTR EQL TEMP_STRING + .OBJECT_FAB[FAB$B_FNS] - 1)
783 THEN
784 BEGIN
```

```
785 0782 5      FLAGS[IN_ELLIPSE] = 1;      ! Show that there's a trailing ellipse
786 0783 5      CH$WCHAR(,STR_PTR,STR_PTR - 3); ! Put the end bracket in place
787 0784 5      OBJECT_FAB[FAB$C_FNA] = TEMP_STRING; ! Set up FAB fields
788 0785 5      OBJECT_FAB[FAB$B_FNS] = .STR_PTR - 3 - TEMP_STRING + 1;
789 0786 4      END;
790 0787 4      END
791 0788 4
792 0789 4      ! If here, then the trailing ellipse has been processed, and this is the
793 0790 4      ! second time thru. Restore the original file name.
794 0791 4
795 0792 4      ELSE
796 0793 4          BEGIN
797 0794 4              OBJECT_FAB[FAB$C_FNA] = .FILE_DESC[DSC$A_POINTER]; ! Original filename
798 0795 4              OBJECT_FAB[FAB$B_FNS] = .FILE_DESC[DSC$W_LENGTH]; ! Original length
799 0796 4              FLAGS[IN_ELLIPSE] = 0; ! Ellipse processed
800 0797 4          END;
801 0798 4
802 0799 4      ! Parse the input string
803 0800 4
804 0801 4      $NAM_INIT (NAM = RELATED_NAM); ! Re-init the RLF
805 0802 4      IF (.OBJECT_NAM[NAM$B_DEV] NEQ 0) ! If a device was
806 0803 4      THEN ! specified, then
807 0804 4          BEGIN
808 0805 4              OBJECT_FAB[FAB$C_DNA] = .OBJECT_NAM[NAM$C_DEV]; ! Make device sticky
809 0806 4              OBJECT_FAB[FAB$B_DNS] = .OBJECT_NAM[NAM$B_DEV];
810 0807 4          END;
811 0808 4      IF NOT (STATUS = $PARSE (FAB = OBJECT_FAB))
812 0809 4      THEN
813 0810 4          BEGIN
814 0811 4              DESC[DSC$W_LENGTH] = .OBJECT_FAB[FAB$B_FNS];
815 0812 4              DESC[DSC$A_POINTER] = .OBJECT_FAB[FAB$C_FNA];
816 0813 4              FILE_ERROR (SET$SYNTAX, OBJECT_FAB, .STATUS, 0);
817 0814 4          END;
818 0815 4
819 0816 4      ! Check the parsed string for legality, i.e. nothing after the directory
820 0817 4
821 0818 4      IF (.OBJECT_NAM[NAM$B_NAME] NEQ 0 OR
822 0819 4          .OBJECT_NAM[NAM$B_TYPE] NEQ 1 OR
823 0820 4          .OBJECT_NAM[NAM$B_VER] NEQ 1 )
824 0821 4      THEN
825 0822 4          BEGIN
826 0823 4              DESC[DSC$W_LENGTH] = .OBJECT_FAB[FAB$B_FNS];
827 0824 4              DESC[DSC$A_POINTER] = .OBJECT_FAB[FAB$C_FNA];
828 0825 4              FILE_ERROR (SET$SYNTAX, OBJECT_FAB, SS$BADIRECTORY, 0);
829 0826 4          END;
830 0827 4
831 0828 4      ! Determine what the directory terminator character was, and save it.
832 0829 4
833 0830 4      ENDCHAR = .(.OBJECT_NAM[NAM$C_DIR] + .OBJECT_NAM[NAM$B_DIR] - 1);
834 0831 4
835 0832 4      ! The directory string must now be analyzed and manipulated so that the
836 0833 4      ! final directory entry becomes a file. First, initialize some pointers.
837 0834 4
838 0835 4      DESC[DSC$W_LENGTH] = .OBJECT_NAM[NAM$B_ESL] - 2;
839 0836 4      DESC[DSC$A_POINTER] = .OBJECT_NAM[NAM$C_ESA];
840 0837 4      STR_PTR = .DESC[DSC$A_POINTER];
841 0838 4      STR_LEN = .DESC[DSC$W_LENGTH];
```



```

842 0839 PTR = 0;
843 0840 EOS = .DESC[DSC$A_POINTER] + .DESC[DSC$W_LENGTH] - 1;
844 0841
845 0842 ! Look for wildcard ellipses
846 0843
847 0844 WHILE NOT CH$FAIL (TEMP = CH$FIND_SUB (.STR_LEN, .STR_PTR,
848 0845 3, UPLIT ('...')))
849 0846 DO
850 0847 BEGIN
851 0848
852 0849 ! Make PTR point to the beginning of the "...", and advance the string
853 0850 ! pointer to the character just past the "...".
854 0851
855 0852 PTR = .TEMP;
856 0853 STR_LEN = .STR_LEN - (.TEMP - .STR_PTR) - 3;
857 0854 STR_PTR = .TEMP + 3;
858 0855 END;
859 0856
860 0857 ! If there was any occurrence of "...", point just past it.
861 0858
862 0859 IF .PTR NEQ 0 THEN PTR = .PTR + 3;
863 0860
864 0861 ! Find the last directory in the specification
865 0862
866 0863 WHILE NOT CH$FAIL (TEMP = CH$FIND_CH (.STR_LEN, .STR_PTR, '.'))
867 0864 DO
868 0865 BEGIN
869 0866
870 0867 ! Make PTR point to the ".", and advance the string pointer to
871 0868 ! the first character after the "."
872 0869
873 0870 PTR = .TEMP;
874 0871 STR_LEN = .STR_LEN - (.TEMP - .STR_PTR) - 1;
875 0872 STR_PTR = .TEMP + 1;
876 0873 END;
877 0874
878 0875 IF .PTR NEQ 0
879 0876 THEN
880 0877 BEGIN
881 0878
882 0879 ! If here, then either a trailing ellipse, or a final sub-directory
883 0880 ! was specified. If the pointer is at the bracket, then there is a
884 0881 ! trailing ellipse, in which case only a "*" is required.
885 0882
886 0883 IF .PTR EQL .EOS
887 0884 THEN
888 0885 BEGIN
889 0886 CH$A_MCHAR ('*', PTR); ! Stick an asterisk after the bracket.
890 0887 PTR = .PTR + 1; ! Adjust the pointer.
891 0888 END
892 0889
893 0890 ! If the pointer is inside the bracket, then the last directory name
894 0891 ! must be moved out of the brackets.
895 0892
896 0893 ELSE
897 0894 BEGIN
898 0895
```

```
0899 0896 5 ! Check to see if the directory is [main.sub] or [main...sub]
0900 0897 5
0901 0898 5 IF .PTR EQLU .STR_PTR
0902 0899 5 THEN
0903 0900 6 BEGIN ! [main...sub] form
0904 0901 6 STR_LEN = .EOS - .PTR;
0905 0902 6 CH$MOVE (.STR_LEN, .PTR, .PTR+1);
0906 0903 6 CH$WCHAR (.ENDCHAR, .PTR);
0907 0904 6 PTR = .PTR + .STR_LEN + 1;
0908 0905 6 END ! end of [main...sub] processing
0909 0906 5 ELSE
0910 0907 6 BEGIN ! [main.sub] form
0911 0908 6 STR_LEN = .EOS - .STR_PTR;
0912 0909 6 CH$WCHAR A (.ENDCHAR, .PTR);
0913 0910 6 PTR = .PTR + .STR_LEN;
0914 0911 5 END; ! end of [main.sub] processing
0915 0912 4 END; ! End of non-zero pointer stuff
0916 0913 4 ELSE
0917 0914 3 BEGIN
0918 0915 4
0919 0916 4 ! If the pointer is still zero, then there is either a wildcard, a main
0920 0917 4 ! directory, or a [g,m] directory. In all such cases, a main directory
0921 0918 4 ! of [000000] must be fabricated.
0922 0919 4
0923 0920 4 STATUS = CH$FIND_CH (.STR_LEN, .STR_PTR, ','); ! Save for later
0924 0921 4
0925 0922 4 ! Move the string out seven spaces and insert "000000]"
0926 0923 4
0927 0924 4 STR_PTR = .DESC[DSC$A_POINTER] + .OBJECT NAM[NAM$B_DEV] + 1;
0928 0925 4 TEMP = CH$MOVE (.EOS - .STR_PTR, .STR_PTR, .STR_PTR + 7);
0929 0926 4 STR_PTR = CH$MOVE (6, UPLIT('000000'), .STR_PTR);
0930 0927 4 CH$MOVE (1, ENDCHAR, .STR_PTR);
0931 0928 4
0932 0929 4 ! If no comma was found, then all that is required is to update the
0933 0930 4 ! pointer.
0934 0931 4
0935 0932 4 IF CH$FAIL (.STATUS) THEN PTR = .TEMP
0936 0933 4
0937 0934 4 ! Otherwise, it's a [g,m] directory. Convert it.
0938 0935 4
0939 0936 4 ELSE
0940 0937 4 BEGIN
0941 0938 5
0942 0939 5 LOCAL TPARSE_BLOCK : $BLOCK[TPASK_LENGTH0]; ! Define a TPARSE block
0943 0940 5
0944 0941 5 CH$FILL (0, TPASK_LENGTH0, TPARSE_BLOCK); ! Zero it.
0945 0942 5 TPARSE_BLOCK[TPASK_COUNT] = TPASK_COUNT0; ! Fill in size
0946 0943 5
0947 0944 5 TPARSE_BLOCK[TPASK_STRINGCNT] = .EOS - .STR_PTR;
0948 0945 5 TPARSE_BLOCK[TPASK_STRINGPTR] = .STR_PTR + 7;
0949 0946 5 IF NOT (STATUS = LIB$TPARSE (TPARSE_BLOCK,
0950 0947 5 DIR_STATE,
0951 0948 5 DIR_KEYS))
0952 0949 5 THEN FILE_ERROR (SET$SYNTAX, OBJECT_FAB, .STATUS, 0)
0953 0950 5 ELSE
0954 0951 5 BEGIN
0955 0952 5
```

```
0953 LOCAL TEMP_DESC : $BLOCK[DSC$C_S_BLN];
0954 TEMP_DESC[DSC$W_LENGTH] = 6;
0955 TEMP_DESC[DSC$A_POINTER] = .STR_PTR + 7;
0956 IF NOT (STATUS = $FAO ($DESCRIPTOR('!2(30W)'),
0957 TEMP_DESC,
0958 TEMP_DESC,
0959 .DIR_GROUP,
0960 .DIR_MEMBER))
0961 THEN FILE_ERROR (SET$SYNTAX, OBJECT_FAB, .STATUS, 0)
0962 ELSE PTR = .STR_PTR + 14;
0963 END;
0964 END;
0965 PTR = CH$MOVE (4, UPLIT ('.DIR'), .PTR);
0966 OBJECT_FAB[FAB$B_FNS] = .PTR - .DESC[DSC$A_POINTER];
0967 OBJECT_FAB[FAB$L_FNA] = .DESC[DSC$A_POINTER];
0968 END;
0969 RETURN 1;
0970 END;
0971
0972
0973
```

! End of routine GET_FILE

```
00 00 30 30 30 30 30 30 002F0 P.ACN: .ASCII \INPUT\
29 57 4F 33 28 32 21 002F5 .BLKB 3
00000005 002F8 P.ACM: .LONG 5
00000000 002FC .ADDRESS P.ACN
00 2E 2E 2E 00300 P.ACO: .ASCII \...\<0>
00 2E 2E 2E 00304 P.ACP: .ASCII \...\<0>
00 00 30 30 30 30 30 00308 P.ACQ: .ASCII \000000\<0><0>
29 57 4F 33 28 32 21 00310 P.ACS: .ASCII \!2(30W)\
00317 .BLKB 1
00000007 00318 P.ACR: .LONG 7
00000000 0031C .ADDRESS P.ACS
52 49 44 2E 00320 P.ACT: .ASCII \.DIR\
```

.PSECT \$OWNS\$,NOEXE,2

00 014AB FILE_DESC:

```
00 014A9 .BYTE 0
00 014AA .BYTE 0
00 014AB .BYTE 0
00 014AC .BYTE 0
00 014AD .BYTE 0
00 014AE .BYTE 0
00 014AF .BYTE 0
```

```
$RMS_PTR=
.EXTRN RELATED NAM
SYSSPARSE, SYSSFAO
```

.PSECT \$CODE\$,NOWRT,2

OFFC 00000 GET_FILE:

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

	F8	AD	0000'	CF	9B	000FD	MOVZBW	OBJECT_FAB+52, DESC	0811
	FC	AD	0000'	CF	D0	00103	MOVL	OBJECT_FAB+44, DESC+4	0812
			04	7E	D4	00109	CLRL	-(SP)	0813
			0000'	AE	DD	0010B	PUSHL	STATUS	
		007710FC	0000'	CF	9F	0010E	PUSHAB	OBJECT_FAB	
0000V	CF		0000'	8F	DD	00112	PUSHL	#7803132	
			0000'	04	FB	00118	CALLS	#4, FILE ERROR	
			0000'	CF	95	0011D	TSTB	OBJECT_NAM+59	0818
	01		0000'	0E	12	00121	BNEQ	12\$	
			0000'	CF	91	00123	CMPB	OBJECT_NAM+60, #1	0819
	01		0000'	07	12	00128	BNEQ	12\$	
			0000'	CF	91	0012A	CMPB	OBJECT_NAM+61, #1	0820
				22	13	0012F	BEQL	13\$	
F8	AD		0000'	CF	9B	00131	MOVZBW	OBJECT_FAB+52, DESC	0823
FC	AD		0000'	CF	D0	00137	MOVL	OBJECT_FAB+44, DESC+4	0824
				7E	D4	0013D	CLRL	-(SP)	0825
	7E		0828	8F	3C	0013F	MOVZWL	#2088, -(SP)	
			0000'	CF	9F	00144	PUSHAB	OBJECT_FAB	
0000V	CF	007710FC	0000'	8F	DD	00148	PUSHL	#7803132	
			0000'	04	FB	0014E	CALLS	#4, FILE ERROR	
	50		0000'	CF	9A	00153	MOVZBL	OBJECT_NAM+58, R0	0830
	50		0000'	CF	C0	00158	ADDL2	OBJECT_NAM+72, R0	
04	AE		FF	A0	90	0015D	MOVB	-1(R0), ENDCHAR	
F8	AD		0000'	CF	9B	00162	MOVZBW	OBJECT_NAM+11, DESC	0835
F8	AD			02	A2	00168	SUBW2	#2, DESC	
FC	AD		0000'	CF	D0	0016C	MOVL	OBJECT_NAM+12, DESC+4	0836
	59		FC	AD	D0	00172	MOVL	DESC+4, R9	0837
	5A			59	D0	00176	MOVL	R9, STR_PTR	
	50		F8	AD	3C	00179	MOVZWL	DESC, R0	0838
	57			50	D0	0017D	MOVL	R0, STR_LEN	
				58	D4	00180	CLRL	PTR	0839
6A		57	0000'	FF	A049	9E	MOVAB	-1(R0)[R9], EOS	0840
				03	39	00187	MATCHC	#3, P.ACP, STR_LEN, (STR_PTR)	0845
				03	13	0018E	BEQL	15\$	
	53			03	D0	00190	MOVL	#3, R3	
				53	D7	00193	DECL	R3	
	5B			73	3E	00195	MOVAB	-(R3), TEMP	
				12	13	00198	BEQL	16\$	
	58			5B	D0	0019A	MOVL	TEMP, PTR	0852
53	5A			5B	C3	0019D	SUBL3	TEMP, STR_PTR, R3	0853
	57		FD	A347	9E	001A1	MOVAB	-3(R3)[STR_LEN], STR_LEN	
	5A		03	AB	9E	001A6	MOVAB	3(R11), STR_PTR	0854
				DB	11	001AA	BRB	14\$	0844
				58	D5	001AC	TSTL	PTR	0859
				03	13	001AE	BEQL	17\$	
	58			03	C0	001B0	ADDL2	#3, PTR	
6A	57			2E	3A	001B3	LOCC	#46, STR_LEN, (STR_PTR)	0863
				02	12	001B7	BNEQ	18\$	
				51	D4	001B9	CLRL	R1	
	5B			51	D0	001BB	MOVL	R1, TEMP	
				12	13	001BE	BEQL	19\$	
	58			5B	D0	001C0	MOVL	TEMP, PTR	0870
53	5A			5B	C3	001C3	SUBL3	TEMP, STR_PTR, R3	0871
	57		FF	A347	9E	001C7	MOVAB	-1(R3)[STR_LEN], STR_LEN	
	5A		01	AB	9E	001CC	MOVAB	1(R11), STR_PTR	0872
				E1	11	001D0	BRB	17\$	0863
				58	D5	001D2	TSTL	PTR	0875

			56		32	13	001D4	BEQL	22\$			
					58	D1	001D6	CMPL	PTR, EOS			0883
					07	12	001D9	BNEQ	20\$			
			88		58	D6	001DB	INCL	PTR			0886
					2A	90	001DD	MOVB	#42, (PTR)+			
			5A		58	11	001E0	BRB	24\$			0883
					58	D1	001E2	CMPL	PTR, STR_PTR			0898
					14	12	001E5	BNEQ	21\$			
			56		58	C3	001E7	SUBL3	PTR, EOS, STR_LEN			0901
01	57		68		57	28	001EB	MOVCL	STR_LEN, (PTR), 1(PTR)			0902
	AB		68		AE	90	001F0	MOVB	ENDCHAR, (PTR)			0903
			58		01	A748	9E	001F4	MOVAB	1(STR_LEN)[PTR], PTR		0904
					42	11	001F9	BRB	24\$			0898
	57		56		5A	C3	001FB	SUBL3	STR_PTR, EOS, STR_LEN			0908
			88		04	AE	90	001FF	MOVB	ENDCHAR, (PTR)+		0909
			58		57	C0	00203	ADDL2	STR_LEN, PTR			0910
					35	11	00206	BRB	24\$			0875
	6A		57		2C	3A	00208	LOCC	#44, STR_LEN, (STR_PTR)			0921
					02	12	0020C	BNEQ	23\$			
			6E		51	D4	0020E	CLRL	R1			
			50		0000'	51	D0	00210	MOVL	R1, STATUS		
			5A		01	CF	9A	00213	MOVZBL	OBJECT_NAM+57, R0		0925
			56			A049	9E	00218	MOVAB	1(R0)[R9], STR_PTR		
07	50		6A			5A	C3	0021D	SUBL3	STR_PTR, EOS, R0		0926
	AA		5B			50	28	00221	MOVCL	R0, (STR_PTR), 7(STR_PTR)		
	6A	0000'	CF			53	D0	00226	MOVL	R3, TEMP-		
			5A			06	28	00229	MOVCL	#6, P.ACR, (STR_PTR)		0927
			6A			53	D0	0022F	MOVL	R3, STR_PTR		
					04	AE	90	00232	MOVB	ENDCHAR, (STR_PTR)		0928
						6E	D5	00236	TSTL	STATUS		0933
			58			05	12	00238	BNEQ	25\$		
						5B	D0	0023A	MOVL	TEMP, PTR		
			6E			11	0023D	BRB	28\$			
24	00		6E			00	2C	0023F	MOVCL	#0, (SP), #0, #36, TPARSE_BLOCK		0942
					10	AE		00244				
						08	D0	00246	MOVL	#8, TPARSE_BLOCK		0943
18	AE	10	56			5A	C3	0024A	SUBL3	STR_PTR, EOS, TPARSE_BLOCK+8		0945
			53			AA	9E	0024F	MOVAB	7(R0), R3		0946
		1C	AE			53	D0	00253	MOVL	R3, TPARSE_BLOCK+12		
					0000'	CF	9F	00257	PUSHAB	DIR_KEYS		0947
					0000'	CF	9F	0025B	PUSHAB	DIR_STATE		
					18	AE	9F	0025F	PUSHAB	TPARSE_BLOCK		
		00000000G	00			03	FB	00262	CALLS	#3, LIB\$TPARSE		
			6E			50	D0	00269	MOVL	R0, STATUS		
			24			6E	E9	0026C	BLBC	STATUS, 26\$		0954
		08	AE			06	B0	0026F	MOVW	#6, TEMP_DESC		0955
		0C	AE			53	D0	00273	MOVL	R3, TEMP_DESC+4		
			7E			CF	7D	00277	MOVQ	DIR_GROUP, -(SP)		0960
					0000'	AE	9F	0027C	PUSHAB	TEMP_DESC		
					14	AE	9F	0027F	PUSHAB	TEMP_DESC		
					0000'	CF	9F	00282	PUSHAB	P.ACR		
		00000000G	00			05	FB	00286	CALLS	#5, SYS\$FAO		
			6E			50	D0	0028D	MOVL	R0, STATUS		
			16			6E	E8	00290	BLBS	STATUS, 27\$		
						7E	D4	00293	CLRL	-(SP)		0961
					04	AE	DD	00295	PUSHL	STATUS		
					0000'	CF	9F	00298	PUSHAB	OBJECT_FAB		

AED\$SETACL
V04-000

D 16
16-Sep-1984 00:02:30
14-Sep-1984 11:52:34

VAX-11 BL1sy-32 V4.0-742
[ACLEDT.SRC]SETACL.B32;1

Page 37
(4)

0000V	CF	007710FC	8F	DD	0029C	PUSHL	#7803132	
			04	FB	002A2	CALLS	#4, FILE_ERROR	
			04	11	002A7	BRB	28\$	
	58	OE	AA	9E	002A9	MOVAB	14(R10), PTR	0962
0000' CF	88	0000'	CF	D0	002AD	MOVL	P.ACT, (PTR)+	0966
	58		59	B3	002B2	SUBB3	R9, PTR, OBJECT_FAB+52	0967
	CF		59	D0	002B8	MOVL	R9, OBJECT_FAB+44	0968
	50		01	D0	002BD	MOVL	#1, R0	0971
				04	002C0	RET		
			50	D4	002C1	CLRL	R0	0973
			04	04	002C3	RET		

; Routine Size: 708 bytes, Routine Base: \$CODE\$ + 07E7

```
0974 1 ROUTINE PROCESS_FILE =
0975 1 ++
0976 1
0977 1 FUNCTIONAL DESCRIPTION:
0978 1
0979 1     This routine takes the spec from LIB$FILE_SCAN, and calls the
0980 1     appropriate routine based upon the command line qualifiers.
0981 1
0982 1 CALLING SEQUENCE:
0983 1     PROCESS_FILE
0984 1
0985 1 INPUT PARAMETERS:
0986 1     none
0987 1
0988 1 IMPLICIT INPUTS:
0989 1     none
0990 1
0991 1 OUTPUT PARAMETERS:
0992 1     none
0993 1
0994 1 IMPLICIT OUTPUTS:
0995 1     none
0996 1
0997 1 ROUTINE VALUE:
0998 1     1 if successful
0999 1     error code otherwise
1000 1
1001 1 SIDE EFFECTS:
1002 1     none
1003 1
1004 1 --
1005 1
1006 1 BEGIN
1007 1
1008 1 LOCAL
1009 1
1010 1     FILE_NAME      : $BLOCK [DSC$C_S_BLN],      ! File name to log
1011 1     FAB             : $FAB_DECL,                ! Storage for the FAB
1012 1     NAM             : $NAM_DECL,                ! Storage for the NAME block
1013 1     XABDAT          : $XABDAT_DECL,              ! Date XAB storage
1014 1     XABPRO          : $XABPRO_DECL,              ! Protection XAB storage
1015 1     FILE_CHAR       : $BLOCK [4],                ! Target file characteristics
1016 1     IO_STATUS       : VECTOR [4, WORD],          ! I/O status block
1017 1     STATUS          :                          ! Local routine return status
1018 1     STATUS1         :                          ! Second local routine exit status
1019 1
1020 1 ! Open the the specified file.
1021 1
1022 1 CH$FILL (0, 3 * ITM$S_ITEM, ATR ARGLIST);
1023 1 CH$MOVE (NAM$C_BLN, .OBJECT_FAB[FAB$L_NAM], NAM);
1024 1 $FAB_INIT (FAB = FAB,
1025 1           FAC = <GET, PUT>,
1026 1           FOP = <NAM, UFO>,
1027 1           NAM = NAM,
1028 1           SHR = NIL,
1029 1           XAB = XABDAT);
1030 1
1031 1 $XABDAT_INIT (XAB = XABDAT,
```

```
1035 1031      NXT = XABPRO);
1036 1032 $XABPRO_INIT (XAB = XABPRO);
1037 1033
1038 1034 STATUS = $OPEN (FAB = FAB);
1039 1035
1040 1036 ! Set up the actual file name.
1041 1037
1042 1038 CH$FILL (0, DSC$S_BLN, FILE_NAME);
1043 1039 IF .NAM[NAM$B_RSL] NEQ 0
1044 1040 THEN
1045 1041     BEGIN
1046 1042         FILE_NAME[DSC$W_LENGTH] = .NAM[NAM$B_RSL];
1047 1043         FILE_NAME[DSC$A_POINTER] = .NAM[NAM$C_RSA];
1048 1044     END
1049 1045 ELSE IF .NAM[NAM$B_ESL] NEQ 0
1050 1046 THEN
1051 1047     BEGIN
1052 1048         FILE_NAME[DSC$W_LENGTH] = .NAM[NAM$B_ESL];
1053 1049         FILE_NAME[DSC$A_POINTER] = .NAM[NAM$C_ESA];
1054 1050     END
1055 1051 ELSE
1056 1052     BEGIN
1057 1053         FILE_NAME[DSC$W_LENGTH] = .FAB[FAB$B_FNS];
1058 1054         FILE_NAME[DSC$A_POINTER] = .FAB[FAB$C_FNA];
1059 1055     END;
1060 1056
1061 1057 ! If there are any errors on the open, note them.
1062 1058
1063 1059 IF NOT .STATUS
1064 1060 THEN
1065 1061     BEGIN
1066 1062
1067 1063 ! If the error is a "file locked by another user" error and the file-id of the
1068 1064 ! source and target files match, simply ignore the error and go process the next
1069 1065 ! in line. Otherwise, note the error.
1070 1066
1071 1067     IF .FAB[FAB$L_STS] NEQ RMS$ FLK
1072 1068     OR CH$NEQ (6, SUBJECT_NAM[NAM$W_FID], 6, OBJECT_NAM[NAM$W_FID], 0)
1073 1069     THEN FILE_ERROR (SET$OPENIN, FAB, .FAB[FAB$L_STS], .FAB[FAB$L_STV]);
1074 1070     RETURN 1;
1075 1071     END;
1076 1072
1077 1073 CHAN = .FAB[FAB$L_STV];
1078 1074
1079 1075 ! See if the file matches the criteria specified by the common command
1080 1076 ! qualifiers.
1081 1077
1082 1078 IF NOT LIB$QUAL_FILE_MATCH (COMMON_CTX,
1083 1079     FAB,
1084 1080     0,
1085 1081     $DESCRIPTOR ('%SET-I-MODIFY, modify ACL on !AS [N]:'),
1086 1082     %REF (FILE_NAME),
1087 1083     0) THEN RETURN 1;
1088 1084
1089 1085 ! Determine whether or not the target file is a directory file.
1090 1086
1091 1087 ATR_ARGLIST[0, ITMSW_ITMCO] = ATR$C_UCHAR;
```



```
1092 1088 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ATR$S_UCHAR;
1093 1089 ATR_ARGLIST[0, ITMSW_BUFADR] = FILE_CHAR;
1094 1090 STATUS = $QIOW (CHAN = .CHAN,
1095 1091 FUNC = IOS_ACCESS,
1096 1092 IOSB = IO_STATUS,
1097 1093 PS = ATR_ARGLIST);
1098 1094 IF .STATUS THEN STATUS = .IO_STATUS[0];
1099 1095 IF NOT .STATUS
1100 1096 THEN
1101 1097 BEGIN
1102 1098 SIGNAL (SETS_OPENIN, 1, FILE_NAME, .STATUS, 0);
1103 1099 RETURN 1; ! Return without doing anything
1104 1100 END;
1105 1101 FLAGS[DIRECTORY] = .FILE_CHAR[FCH$V_DIRECTORY];
1106 1102
1107 1103 ! If the /DEFAULT qualifier is being processed, make sure that the parent
1108 1104 ! directory of the current file is accessed on the source object channel.
1109 1105
1110 1106 IF .FLAGS[QUAL_DEFAULT]
1111 1107 THEN
1112 1108 BEGIN
1113 1109
1114 1110 ! If a channel has not been assigned to the source object, assign a channel
1115 1111 ! to the device for the parent directory.
1116 1112
1117 1113 IF .SCHAN EQL 0
1118 1114 THEN
1119 1115 BEGIN
1120 1116 CH$FILL (0, DSC$C_S_BLN, SDEVICE_DESC);
1121 1117 SDEVICE_DESC[DSC$S_LENGTH] = .VECTOR [NAM[NAM$T_DVI], 0;, BYTE];
1122 1118 SDEVICE_DESC[DSC$A_POINTER] = NAM[NAM$T_DVI] + T;
1123 1119 STATUS = $ASSIGN (DEVNAM = SDEVICE_DESC, CHAN = .SCHAN);
1124 1120 IF NOT .STATUS
1125 1121 THEN
1126 1122 BEGIN
1127 1123 SIGNAL (SETS_OPENIN, 1, SDEVICE_DESC, .STATUS, 0);
1128 1124 RETURN 1;
1129 1125 END;
1130 1126 END;
1131 1127
1132 1128 ! If there is already a directory accessed on the source object channel, and
1133 1129 ! the file-IDs are not the same, deaccess the directory file.
1134 1130
1135 1131 IF .SFILE_FIB[FIB$W_FID_NUM] NEQ 0
1136 1132 AND CH$NEQ (FIB$S_FID, SFILE_FIB[FIB$W_FID], FIB$S_FID, NAM[NAM$W_DID], 0)
1137 1133 THEN
1138 1134 BEGIN
1139 1135 STATUS = $QIOW (CHAN = .SCHAN,
1140 1136 FUNC = IOS_DEACCESS,
1141 1137 IOSB = IO_STATUS);
1142 1138 IF .STATUS THEN STATUS = .IO_STATUS[0];
1143 1139 IF NOT .STATUS THEN SIGNAL (SETS_CLOSEIN, 1, SUBJECT_DESC, .STATUS, 0);
1144 1140 SFILE_FIB[FIB$W_FID_NUM] = 0; ! To force access below
1145 1141
1146 1142 ! Now release the read lock that was taken out for the directory file.
1147 1143
1148 1144 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_UNLOCK_ACL;
```

```
1145 4 ATR_ARGLIST[0, ITMSW_BUFSIZ] = 4;
1146 4 ATR_ARGLIST[0, ITMSL_BUFADR] = SACL_LOCKID;
1147 4 STATUS = $CHANGE_ACL (CHAN = .SCHAN,
1148 4 OBJTYP = SUBJECT_TYPE,
1149 4 OBJNAM = SUBJECT_DESC,
1150 4 ITMLST = ATR_ARGLIST);
1151 4 IF NOT .STATUS THEN SIGNAL (SET$_CLOSEIN, 1, SUBJECT_DESC, .STATUS, 0);
1152 4 END;
1153 4
1154 4 ! If there is not a directory file currently accessed, do so now.
1155 4
1156 4 IF .SFILE_FIB[FIB$W_FID_NUM] EQL 0
1157 4 THEN
1158 4 BEGIN
1159 4 SFILE_FIB[FIB$W_FID] = 0;
1160 4 CH$MOVE (FIB$S_FID, NAM[NAM$W_DID], SFILE_FIB[FIB$W_FID]);
1161 4 STATUS = $QIOW (CHAN = .SCHAN,
1162 4 FUNC = IOS_ACCESS OR IOSM_ACCESS,
1163 4 IOSB = IO_STATUS,
1164 4 P1 = SFILE_DESC);
1165 4 IF .STATUS THEN STATUS = .IO_STATUS[0];
1166 4 IF NOT .STATUS
1167 4 THEN
1168 4 BEGIN
1169 4 SIGNAL (SET$_OPENIN, 1, SDEVICE_DESC, .STATUS, 0);
1170 4 RETURN 1;
1171 4 END;
1172 4
1173 4 ! Get the file spec for the parent directory file, in case any errors occur.
1174 4
1175 4 LIB$FID_TO_NAME (SDEVICE_DESC, SFILE_FIB[FIB$W_FID],
1176 4 SUBJECT_DESC, SUBJECT_DESC,
1177 4 0, STATOS1);
1178 4
1179 4 ! Attempt to obtain a read lock for the source object.
1180 4
1181 4 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_RLOCK_ACL;
1182 4 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_RLOCK_ACL;
1183 4 ATR_ARGLIST[0, ITMSL_BUFADR] = SACL_LOCKID;
1184 4 STATUS = $CHANGE_ACL (CHAN = .SCHAN,
1185 4 OBJTYP = SUBJECT_TYPE,
1186 4 OBJNAM = SUBJECT_DESC,
1187 4 ITMLST = ATR_ARGLIST);
1188 4
1189 4 IF NOT .STATUS
1190 4 THEN
1191 4 BEGIN
1192 4 IF .STATUS EQL SS$ NOTQUEUED
1193 4 THEN SIGNAL (SET$ OBJLOCKED)
1194 4 ELSE SIGNAL (.STATUS);
1195 4 RETURN 1;
1196 4 END;
1197 4
1198 4 END;
1199 4
1200 4 END;
1201 4
1202 4 ! Attempt to obtain a write lock for the target object.
1203 4
1204 4 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_WLOCK_ACL;
```

```
1206 1202 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_WLOCK_ACL;
1207 1203 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACL_LOCKID;
1208 1204 2 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1209 1205 2 OBJTYP = OBJECT_TYPE,
1210 1206 2 OBJNAM = FILE_NAME,
1211 1207 2 ITMLST = ATR_ARGLIST);
1212 1208 2 IF NOT .STATUS
1213 1209 2 THEN
1214 1210 2 BEGIN
1215 1211 2 IF .STATUS EQL $$ NOTQUEUED
1216 1212 2 THEN SIGNAL (SET$OBJLOCKED)
1217 1213 2 ELSE SIGNAL (.STATUS);
1218 1214 2 RETURN 1;
1219 1215 2 END;
1220 1216 2 ! Call the necessary routine based upon the command line qualifiers.
1221 1217 2
1222 1218 2 IF .FLAGS[QUAL LIKE] OR .FLAGS[QUAL DEFAULT] THEN STATUS = COPY_ACL (FILE_NAME)
1223 1219 2 ELSE IF .FLAGS[QUAL DELETE] THEN STATUS = DELETE_ACL (FILE_NAME)
1224 1220 2 ELSE IF .FLAGS[QUAL REPLACE] THEN STATUS = REPLACE_ACL (FILE_NAME)
1225 1221 2 ELSE STATUS = ADD_ACL (FILE_NAME);
1226 1222 2
1227 1223 2 ! Now release the write lock that was taken out.
1228 1224 2
1229 1225 2 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_UNLOCK_ACL;
1230 1226 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = 4;
1231 1227 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACL_LOCKID;
1232 1228 2 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1233 1229 2 OBJTYP = OBJECT_TYPE,
1234 1230 2 OBJNAM = FILE_NAME,
1235 1231 2 ITMLST = ATR_ARGLIST);
1236 1232 2
1237 1233 2 ! If logging is being done, indicate that the object has been modified.
1238 1234 2
1239 1235 2 IF .FLAGS[QUAL LOG] AND .STATUS
1240 1236 2 THEN SIGNAL (SET$MODIFIED, 1, FILE_NAME);
1241 1237 2
1242 1238 2 ! Tie off the opened input file, if necessary.
1243 1239 2
1244 1240 2 IF .STATUS
1245 1241 2 THEN
1246 1242 2 BEGIN
1247 1243 2 STATUS = $QIOW (CHAN = .CHAN,
1248 1244 2 FUNC = IO$DEACCESS,
1249 1245 2 IOSB = IO_STATUS);
1250 1246 2 IF .STATUS THEN STATUS = IO_STATUS[0];
1251 1247 2 IF NOT .STATUS
1252 1248 2 THEN
1253 1249 2 BEGIN
1254 1250 2 FILE_ERROR (SET$CLOSEIN, FAB, .STATUS, 0);
1255 1251 2 RETURN 1;
1256 1252 2 END;
1257 1253 2 STATUS = $DASSGN (CHAN = .CHAN);
1258 1254 2 IF NOT .STATUS
1259 1255 2 THEN
1260 1256 2 BEGIN
1261 1257 2 FILE_ERROR (SET$CLOSEIN, FAB, .STATUS, 0);
1262 1258 2
```



```
1263      1259      4      RETURN 1;  
1264      1260      3      END;  
1265      1261      2      END;  
1266      1262      1      RETURN 1;  
1267      1263      1      END;  
1268      1264      1      END;  
1269      1265      1      END;
```

! End of routine PROCESS_FILE

```
20 2C 59 46 49 44 4F 4D 2D 49 2D 54 45 53 25 00324 P.ACV: .ASCII \XSET-I-MODIFY, modify ACL on !AS [N]:\ :  
21 20 6E 6F 20 4C 43 41 20 79 66 69 64 6F 6D 00333 :  
3A 5D 4E 5B 20 53 41 00342 :  
00000025 00349 :  
00000000 0034C P.ACU: .BLKB 3 :  
00000000 00350 .LONG 37 :  
00000000 00350 .ADDRESS P.ACV :
```

.EXTRN SYSSASSIGN

.PSECT \$CODE\$,NOWRT,2

OFFC 00000 PROCESS_FILE:

```
5B 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 : 0974  
5A 00000000G 00 9E 00009 MOVAB SET$ OBJLOCKED, R11  
59 00000000G 00 9E 00010 MOVAB SYSSCHANGE ACL, R10  
58 00000000G 00 9E 00017 MOVAB SYSSQ10W, R9  
57 00000000G 00 9E 0001E MOVAB LIBSSIGNAL, R8  
5E 00000000G 00 9E 00023 MOVAB WORST_ERROR, R7  
6E 00000000G 00 2C 00028 MOVAB -336(SP), SP  
00000000G 00 2C 0002D MOVCS #0, (SP), #0, #36, ATR_ARGLIST : 1022  
0050 8F 0098 CE 3C B7 0060 8F 28 00030 MOVCS #96, @OBJECT_FAB+40, NAM : 1023  
6E 00000000G 00 2C 00039 MOVCS #0, (SP), #0, #80, $RMS_PTR : 1029  
A8 AD 5003 8F B0 00042 MOVW #20483, $RMS_PTR  
AC AD 01020000 8F B0 00048 MOVL #16908288, $RMS_PTR+4  
BE AD 2003 8F B0 00050 MOVW #8195, $RMS_PTR+22  
C7 AD 00000000G 02 90 00056 MOVW #2, $RMS_PTR+31  
CC AD 6C AE 9E 0005A MOVAB XABDAT, $RMS_PTR+36  
D0 AD 0098 CE 9E 0005F MOVAB NAM, $RMS_PTR+40  
2C 00 6E 00 2C 00065 MOVCS #0, (SP), #0, #44, $RMS_PTR : 1031  
6C AE 2C12 8F B0 0006C MOVW #11282, $RMS_PTR  
70 AE 14 AE 9E 00072 MOVAB XABPRO, $RMS_PTR+4  
0058 8F 00 6E 00 2C 00077 MOVCS #0, (SP), #0, #88, $RMS_PTR : 1032  
14 AE 14 AE 0007E MOVW #22547, $RMS_PTR  
5813 8F B0 00080 PUSHAB FAB : 1034  
A8 AD 9F 00086 CALLS #1, SYSSOPEN  
00000000G 00 01 FB 00089 MOVL R0, STATUS  
56 50 D0 00090 MOVCS #0, (SP), #0, #8, FILE_NAME : 1038  
08 00 6E 00 2C 00093 MOVZBL NAM+3, R0 : 1039  
F8 50 009B CE 9A 0009A BEQL 1$  
F8 AD 50 B0 000A1 MOVW R0, FILE_NAME : 1042
```

			FC	AD	009C	CE	D0	000A5		MOVL	NAM+4, FILE_NAME+4	1043
						1D	11	000AB		BRB	3\$	1039
				50	00A3	CE	9A	000AD	1\$:	MOVZBL	NAM+11, R0	1045
						0C	13	000B2		BEQL	2\$	
			F8	AD		50	B0	000B4		MOVW	R0, FILE_NAME	1048
			FC	AD	00A4	CE	D0	000B8		MOVL	NAM+12, FILE_NAME+4	1049
						0A	11	000BE		BRB	3\$	1045
			F8	AD	DC	AD	9B	000C0	2\$:	MOVZBW	FAB+52, FILE_NAME	1053
			FC	AD	D4	AD	D0	000C5		MOVL	FAB+44, FILE_NAME+4	1054
				27		56	E8	000CA	3\$:	BLBS	STATUS, 6\$	1059
		0001828A		8F	B0	AD	D1	000CD		CML	FAB+8, #98954	1067
						0D	12	000D5		BNEQ	5\$	
0088	C7	0380	C7			06	29	000D7		CMPC3	#6, SUBJECT_NAM+36, OBJECT_NAM+36	1068
						03	12	000DF		BNEQ	5\$	
				7E	B0	AD	7D	000E4	4\$:	BRW	34\$	
					A8	AD	9F	000E8	5\$:	MOVQ	FAB+8, -(SP)	1069
					0077109A	8F	DD	000EB		PUSHAB	FAB	
						0361	31	000F1		PUSHL	#7803034	
		0324	C7	B4	AD	D0	000F4	6\$:	BRW	33\$		
					7E	D4	000FA		MOVL	FAB+12, CHAN	1073	
		04	AE	F8	AD	9E	000FC		CLRL	-(SP)	1078	
				04	AE	9F	00101		MOVAB	FILE_NAME, 4(SP)	1082	
				0000	CF	9F	00104		PUSHAB	4(SP)		
					7E	D4	00108		PUSHAB	P.ACU	1081	
					A8	AD	9F	0010A		CLRL	-(SP)	1078
				0644	C7	9F	0010D		PUSHAB	FAB		
		00000000G	00		06	FB	00111		PUSHAB	COMMON CTX		
			C6		50	E9	00118		CALLS	#6, LIB\$QUAL_FILE_MATCH		
		0648	C7	00030004	8F	D0	0011B		BLBC	R0, 4\$		
		064C	C7	04	AE	9E	00124		MOVL	#196612, ATR_ARGLIST	1088	
					7E	D4	0012A		MOVAB	FILE_CHAR, ATR_ARGLIST+4	1089	
				0648	C7	9F	0012C		CLRL	-(SP)	1093	
					7E	7C	00130		PUSHAB	ATR_ARGLIST		
					7E	7C	00132		CLRQ	-(SP)		
					7E	7C	00134		CLRQ	-(SP)		
				2C	AE	9F	00136		CLRQ	-(SP)		
					32	DD	00139		PUSHAB	IO STATUS		
				0324	C7	DD	0013B		PUSHL	#50		
					7E	D4	0013F		PUSHL	CHAN		
					0C	FB	00141		CLRL	-(SP)		
			69		50	D0	00144		CALLS	#12, SYS\$QIOW		
			56		56	E9	00147		MOVL	R0, STATUS		
			07		AE	3C	0014A		BLBC	STATUS, 7\$	1094	
			56	0C	56	E8	0014E		MOVZWL	IO STATUS, STATUS		
			0A		7E	D4	00151	7\$:	BLBS	STATUS, 8\$	1095	
					56	DD	00153		CLRL	-(SP)	1098	
				F8	AD	9F	00155		PUSHL	STATUS		
					0139	31	00158		PUSHAB	FILE_NAME		
					05	EF	0015B	8\$:	BRW	18\$		
					50	F0	00161		EXTZV	#5, #1, FILE_CHAR+1, R0	1101	
FD	50	05	AE	01		06	E0	00167	INSV	R0, #2, #1, FLAGS+1		
	A7		03	02		0195	31	0016C	BBS	#6, FLAGS, 9\$	1106	
				FC	A7	C7	D5	0016F	BRW	21\$		
						05EC	30	00173	TSTL	SCHAN	1113	
							12	00175	BNEQ	10\$		
		08	00	6E		05F4	2C	0017A	MOVC5	#0, (SP), #0, #8, SDEVICE_DESC	1116	

		05F4	C7	FF5C	CD	9B	0017D		MOVZBW	NAM+20, SDEVICE_DESC	1117
		05F8	C7	FF5D	CD	9E	00184		MOVAB	NAM+21, SDEVICE_DESC+4	1118
					7E	7C	0018B		CLRQ	-(SP)	1119
				05EC	C7	9F	0018D		PUSHAB	SCHAN	
		00000000G	00	05F4	C7	9F	00191		PUSHAB	SDEVICE_DESC	
			56		04	FB	00195		CALLS	#4, SYSS\$ASSIGN	
			03		50	D0	0019C		MOVL	R0, STATUS	
					56	E8	0019F		BLBS	STATUS, 108	1120
					00E7	31	001A2		BRW	178	
				0608	C7	85	001A5	108:	TSTW	SFILE_FIB+4	1131
					08	13	001A9		BEQL	118	
		FF72	CD	0608	C7	06	29	001AB	CMPC3	#6, SFILE_FIB+4, NAM+42	1132
					03	12	001B3	118:	BNEQ	128	
					0096	31	001B5		BRW	158	
					7E	7C	001B8	128:	CLRQ	-(SP)	1137
					7E	7C	001BA		CLRQ	-(SP)	
					7E	7C	001BC		CLRQ	-(SP)	
					7E	7C	001BE		CLRQ	-(SP)	
				2C	AE	9F	001C0		PUSHAB	IO STATUS	
					34	DD	001C3		PUSHL	#52	
				05EC	C7	DD	001C5		PUSHL	SCHAN	
					7E	D4	001C9		CLRL	-(SP)	
			69		0C	FB	001CB		CALLS	#12, SYSS\$QIOW	
			56		50	D0	001CE		MOVL	R0, STATUS	
			07		56	E9	001D1		BLBC	STATUS, 138	1138
			56		0C	AE	3C	001D4	MOVZWL	IO STATUS, STATUS	
			21		56	E8	001D8		BLBS	STATUS, 148	1139
					7E	D4	001DB	138:	CLRL	-(SP)	
					56	DD	001DD		PUSHL	STATUS	
				0334	C7	9F	001DF		PUSHAB	SUBJECT_DESC	
					01	DD	001E3		PUSHL	#1	
				00771052	8F	DD	001E5		PUSHL	#7802962	
			68		05	FB	001EB		CALLS	#5, LIB\$SIGNAL	
02			03		00	ED	001EE		CMPZV	#0, #3, WORST_ERROR, #2	
					07	18	001F3		BGEQ	148	
			67	10771052	8F	D0	001F5		MOVL	#276238418, WORST_ERROR	
				0608	C7	B4	001FC	148:	CLRQ	SFILE_FIB+4	1140
		0648	C7	000C0004	8F	D0	00200		MOVL	#786436, ATR_ARGLIST	1145
		064C	C7	032C	C7	9E	00209		MOVAB	SACL_LOCKID, ATR_ARGLIST+4	1146
					7E	7C	00210		CLRQ	-(SF)	1150
					7E	D4	00212		CLRL	-(SP)	
				0648	C7	9F	00214		PUSHAB	ATR_ARGLIST	
				0334	C7	9F	00218		PUSHAB	SUBJECT_DESC	
				0330	C7	9F	0021C		PUSHAB	SUBJECT_TYPE	
				05EC	C7	DD	00220		PUSHL	SCHAN	
			6A		07	FB	00224		CALLS	#7, SYSS\$CHANGE_ACL	
			56		50	D0	00227		MOVL	R0, STATUS	
			21		56	E8	0022A		BLBS	STATUS, 158	1151
					7E	D4	0022D		CLRL	-(SP)	
					56	DD	0022F		PUSHL	STATUS	
				0334	C7	9F	00231		PUSHAB	SUBJECT_DESC	
					01	DD	00235		PUSHL	#1	
				00771052	8F	DD	00237		PUSHL	#7802962	
			68		05	FB	0023D		CALLS	#5, LIB\$SIGNAL	
02			03		00	ED	00240		CMPZV	#0, #3, WORST_ERROR, #2	
					07	18	00245		BGEQ	158	
			67	10771052	8F	D0	00247		MOVL	#276238418, WORST_ERROR	

0608	C7	FF72	CD	0604	C7	D4	00257	16\$:	CLRL	SFILE_FIB	1159	
					06	28	00258		MOVC3	#6, NAM+42, SFILE_FIB+4	1160	
					7E	7C	00263		CLRQ	-(SP)	1164	
					7E	7C	00265		CLRQ	-(SP)		
					7E	D4	00267		CLRL	-(SP)		
				05FC	C7	9F	00269		PUSHAB	SFIB_DESC		
					7E	7C	0026D		CLRQ	-(SP)		
				2C	AE	9F	0026F		PUSHAB	IO STATUS		
			7E	72	8F	9A	00272		MOVZBL	#1T4, -(SP)		
				05EC	C7	DD	00276		PUSHL	SCHAN		
					7E	D4	0027A		CLRL	-(SP)		
			69		0C	FB	0027C		CALLS	#12, SYSSQIOW		
			56		50	D0	0027F		MOVL	RO, STATUS		
			07		56	E9	00282		BLBC	STATUS, 17\$	1165	
			56		AE	3C	00285		MOVZWL	IO STATUS, STATUS		
			24		56	E8	00289		BLBS	STATUS, 20\$	1166	
					7E	D4	0028C	17\$:	CLRL	-(SP)	1169	
					56	DD	0028E		PUSHL	STATUS		
				05F4	C7	9F	00290		PUSHAB	SDEVICE_DESC		
					01	DD	00294	18\$:	PUSHL	#1		
				0077109A	8F	DD	00296		PUSHL	#7803034		
			68		05	FB	0029C		CALLS	#5, LIB\$SIGNAL		
02			03		00	ED	0029F		CMPZV	#0, #3, WORST_ERROR, #2		
					07	18	002A4		BGEQ	19\$		
			67	1077109A	8F	D0	002A6		MOVL	#276238490, WORST_ERROR		
					01AA	31	002AD	19\$:	BRW	34\$	1170	
				08	AE	9F	002B0	20\$:	PUSHAB	STATUS1	1175	
					7E	D4	002B3		CLRL	-(SP)		
				0334	C7	9F	002B5		PUSHAB	SOBJECT_DESC		
				0334	C7	9F	002B9		PUSHAB	SOBJECT_DESC		
				0608	C7	9F	002BD		PUSHAB	SFILE_FIB+4		
				05F4	C7	9F	002C1		PUSHAB	SDEVICE_DESC		
					06	FB	002C5		CALLS	#6, LIB\$FID TO NAME		
			00000000G	00	8F	D0	002CC		MOVL	#655364, ATR_ARGLIST	1182	
			0648	C7	000A0004	C7	9E	002D5	MOVAB	SACL_LOCKID, ATR_ARGLIST+4	1183	
			064C	C7	032C	7E	7C	002DC	CLRQ	-(SP)	1187	
					7E	D4	002DE		CLRL	-(SP)		
				0648	C7	9F	002E0		PUSHAB	ATR_ARGLIST		
				0334	C7	9F	002E4		PUSHAB	SOBJECT_DESC		
				0330	C7	9F	002E8		PUSHAB	SOBJECT_TYPE		
				05EC	C7	DD	002EC		PUSHL	SCHAN		
					07	FB	002F0		CALLS	#7, SYSSCHANGE_ACL		
			6A		50	D0	002F3		MOVL	RO, STATUS		
			56		56	E8	002F6		BLBS	STATUS, 21\$	1188	
			000009B8		56	D1	002F9		CMPL	STATUS, #2488	1191	
					35	13	00300		BEQL	22\$		
					55	11	00302		BRB	23\$	1193	
			0648	C7	000B0004	8F	D0	00304	21\$:	MOVL	#720900, ATR_ARGLIST	1202
			064C	C7	04	A7	9E	0030D	MOVAB	ACL_LOCKID, ATR_ARGLIST+4	1203	
					7E	7C	00313		CLRQ	-(SP)	1207	
					7E	D4	00315		CLRL	-(SP)		
				0648	C7	9F	00317		PUSHAB	ATR_ARGLIST		
				F8	AD	9F	0031B		PUSHAB	FILE_NAME		
				08	A7	9F	0031E		PUSHAB	OBJECT_TYPE		

		0324	C7	DD	00321	PUSHL	CHAN		
		6A	07	FB	00325	CALLS	#7, SYSSCHANGE_ACL		
		56	50	DO	00328	MOVL	RO, STATUS		
		4A	56	E8	0032B	BLBS	STATUS, 25\$	1208	
	00000988	8F	56	D1	0032E	CMPL	STATUS, #2488	1211	
			22	12	00335	BNEQ	23\$		
			5B	DD	00337	PUSHL	R11	1212	
		68	01	FB	00339	CALLS	#1, LIB\$SIGNAL		
		50	6B	9E	0033C	MOVAB	SET\$ OBJLOCKED, RO		
		33	50	E8	0033F	BLBS	RO, 24\$		
50	67	50	00000000*	00	9E	00342	MOVAB	<SET\$ OBJLOCKED?>, RO	
		03	00	ED	00349	CMPZV	#0, #3, WORST_ERROR, RO		
			25	18	0034E	BGEQ	24\$		
		67	00000000*	00	9E	00350	MOVAB	<SET\$ OBJLOCKED!268435456>, WORST_ERROR	
			1C	11	00357	BRB	24\$	1211	
			56	DD	00359	PUSHL	STATUS	1213	
		68	01	FB	0035B	CALLS	#1, LIB\$SIGNAL		
		14	56	E8	0035E	BLBS	STATUS, 24\$		
50	56	03	00	EF	00361	EXTZV	#0, #3, STATUS, RO		
50	67	03	00	ED	00366	CMPZV	#0, #3, WORST_ERROR, RO		
			08	18	0036B	BGEQ	24\$		
	67	56	10000000	8F	C9	0036D	BISL3	#268435456, STATUS, WORST_ERROR	
			00E2	31	00375	BRW	34\$	1214	
05	FC	A7		02	E0	00378	BBS	#2, FLAGS, 26\$	1219
0A	FC	A7		06	E1	0037D	BBC	#6, FLAGS, 27\$	
		F8	AD	9F	00382	PUSHAB	FILE_NAME		
	0000V	CF		01	FB	00385	CALLS	#1, COPY_ACL	
				26	11	0038A	BRB	30\$	
0A	FC	A7		01	E1	0038C	BBC	#1, FLAGS, 28\$	1220
		F8	AD	9F	00391	PUSHAB	FILE_NAME		
	0000V	CF		01	FB	00394	CALLS	#1, DELETE_ACL	
				17	11	00399	BRB	30\$	
0A	FC	A7		04	E1	0039B	BBC	#4, FLAGS, 29\$	1221
		F8	AD	9F	003A0	PUSHAB	FILE_NAME		
	0000V	CF		01	FB	003A3	CALLS	#1, REPLACE_ACL	
				08	11	003A8	BRB	30\$	
		F8	AD	9F	003AA	PUSHAB	FILE_NAME	1222	
	0000V	CF		01	FB	003AD	CALLS	#1, ADD_ACL	
				50	DO	003B2	MOVL	RO, STATUS	
0648	C7	000C0004		8F	DO	003B5	MOVL	#786436, ATR_ARGLIST	1227
064C	C7	04		A7	9E	003BE	MOVAB	ACL_LOCKID, ATR_ARGLIST+4	1228
				7E	7C	003C4	CLRQ	-(SP)	1232
				7E	D4	003C6	CLRL	-(SP)	
		0648	C7	9F	003C8	PUSHAB	ATR_ARGLIST		
		F8	AD	9F	003CC	PUSHAB	FILE_NAME		
		08	A7	9F	003CF	PUSHAB	OBJECT_TYPE		
		0324	C7	DD	003D2	PUSHL	CHAN		
		6A	07	FB	003D6	CALLS	#7, SYSSCHANGE_ACL		
		56	50	DO	003D9	MOVL	RO, STATUS		
30	FC	A7		03	E1	003DC	BBC	#3, FLAGS, 31\$	1236
		76		56	E9	003E1	BLBC	STATUS, 34\$	
			F8	AD	9F	003E4	PUSHAB	FILE_NAME	1237
				01	DD	003E7	PUSHL	#1	
		00000000G	00	9F	003E9	PUSHAB	SET\$ MODIFIED		
		68	03	FB	003EF	CALLS	#3, LIB\$SIGNAL		
		50	00000000G	00	9E	003F2	MOVAB	SET\$ MODIFIED, RO	
		15		50	E8	003F9	BLBS	RO, 31\$	

50	67	50 00000000*	00 9E 003FC	MOVAB	<SETS_MODIFIED&7>, R0	
		03	00 ED 00403	CMPZV	#0, #3, WORST_ERROR, R0	
			07 18 00408	BGEQ	31\$	
		67 00000000*	00 9E 0040A	MOVAB	<SETS_MODIFIED!268435456>, WORST_ERROR	
		46	56 E9 00411	BLBC	STATUS, 34\$	1241
			7E 7C 00414	CLRQ	-(SP)	1246
			7E 7C 00416	CLRQ	-(SP)	
			7E 7C 00418	CLRQ	-(SP)	
			7E 7C 0041A	CLRQ	-(SP)	
		2C	AE 9F 0041C	PUSHAB	IO STATUS	
			34 DD 0041F	PUSHL	#52	
		0324	C7 DD 00421	PUSHL	CHAN	
			7E 04 00425	CLRL	-(SP)	
		69	0C FB 00427	CALLS	#12, SYSSQIOW	
		56	50 D0 0042A	MOVL	R0, STATUS	
		18	56 E9 0042D	BLBC	STATUS, 32\$	1247
		56 0C	AE 3C 00430	MOVZWL	IO STATUS, STATUS	
		11	56 E9 00434	BLBC	STATUS, 32\$	1248
		0324	C7 DD 00437	PUSHL	CHAN	1254
		00000000G	00 01 FB 0043B	CALLS	#1, SYSSDASSGN	
			50 D0 00442	MOVL	R0, STATUS	
			56 E8 00445	BLBS	STATUS, 34\$	1255
			7E D4 00448	CLRL	-(SP)	1258
			56 DD 0044A	PUSHL	STATUS	
		A8	AD 9F 0044C	PUSHAB	FAB	
		00771052	8F DD 0044F	PUSHL	#7802962	
		0000V	CF 04 FB 00455	CALLS	#4, FILE_ERROR	
			50 01 D0 0045A	MOVL	#1, R0	1263
			04 0045D	RET		1265

; Routine Size: 1118 bytes, Routine Base: \$CODE\$ + 0AAB


```
1271 1266 1 ROUTINE ADD_ACL (OBJECT_NAME_DESC) =
1272 1267 1
1273 1268 1 ++
1274 1269 1
1275 1270 1 FUNCTIONAL DESCRIPTION:
1276 1271 1
1277 1272 1 This routine adds ACEs to the end of the ACL or inserts ACEs into
1278 1273 1 various points within the ACL.
1279 1274 1
1280 1275 1 CALLING SEQUENCE:
1281 1276 1 ADD_ACL (ARG1)
1282 1277 1
1283 1278 1 INPUT PARAMETERS:
1284 1279 1 ARG1: address of the FAB
1285 1280 1
1286 1281 1 IMPLICIT INPUTS:
1287 1282 1 none
1288 1283 1
1289 1284 1 OUTPUT PARAMETERS:
1290 1285 1 none
1291 1286 1
1292 1287 1 IMPLICIT OUTPUTS:
1293 1288 1 none
1294 1289 1
1295 1290 1 ROUTINE VALUE:
1296 1291 1 1 if successful
1297 1292 1 error code otherwise
1298 1293 1
1299 1294 1 SIDE EFFECTS:
1300 1295 1 none
1301 1296 1
1302 1297 1 --
1303 1298 1
1304 1299 2 BEGIN
1305 1300 2
1306 1301 2 LOCAL
1307 1302 2 STATUS; ! Local routine return status
1308 1303 2
1309 1304 2 ! Preset the context to start adding ACEs at the beginning of the ACL.
1310 1305 2
1311 1306 2 ACL_CONTEXT = 0;
1312 1307 2
1313 1308 2 ! If this is a new ACL, delete any ACL that currently exists on the object.
1314 1309 2
1315 1310 2 IF .FLAGS[QUAL_NEW]
1316 1311 2 THEN
1317 1312 3 BEGIN
1318 1313 3 ATR_ARGLIST[0, ITMSW_ITMCO] = ACL$C_DELETEACL;
1319 1314 3 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_DELETEACL;
1320 1315 3 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1321 1316 3 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1322 1317 3 OBJTYP = OBJECT TYPE,
1323 1318 3 OBJNAM = OBJECT NAME_DESC,
1324 1319 3 ITMLST = ATR_ARGLIST,
1325 1320 3 CONTXT = ACL_CONTEXT);
1326 1321 3 IF NOT .STATUS
1327 1322 3 THEN
```

P
P
P
P

```
1328      BEGIN
1329      SIGNAL (SET$WRITEERR, .OBJECT_NAME_DESC, .STATUS, 0);
1330      RETURN SET$WRITEERR OR ST$SM_INHIB_MSG;
1331      END;
1332  END;
1333
1334      ! For an insert, first locate the ACE after which the new ACEs will be added.
1335
1336      IF .FLAGS[QUAL_AFTER]
1337      THEN
1338      BEGIN
1339      ACE_POINTER = .OLD_ACE_HEAD[ACEQ_L_FLINK];
1340      CH$MOVE (.S$BLOCK[ACE_POINTER[ACEQ_T_ACE], ACE$B_SIZE],
1341      ACE_POINTER[ACEQ_T_ACE], ACE);
1342      ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_FNDACLENT;
1343      ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACE$B_SIZE];
1344      ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1345      STATUS = $CHANGE_ACL (CHAN = .CHAN,
1346      OBJTYP = OBJECT_TYPE,
1347      OBJNAM = .OBJECT_NAME_DESC,
1348      ITMLST = ATR_ARGLIST,
1349      CONTXT = ACL_CONTEXT);
1350
1351      IF NOT .STATUS
1352      THEN
1353      BEGIN
1354      SIGNAL (SET$WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1355      RETURN SET$WRITEERR OR ST$SM_INHIB_MSG;
1356      END;
1357      IF .ACE[ACE$B_SIZE] EQL 0
1358      THEN
1359      BEGIN
1360      IF .ACE[ACE$W_FLAGS] NEQ SS$_ACLEMPY
1361      THEN
1362      BEGIN
1363      SIGNAL (SET$WRITEERR, 1, .OBJECT_NAME_DESC, .ACE[ACE$W_FLAGS], 0);
1364      RETURN SET$WRITEERR OR ST$SM_INHIB_MSG;
1365      END;
1366      END;
1367      ACL_CONTEXT = .ACL_CONTEXT + 1;
1368      END;
1369
1370      ! Now that the context has been set, add the new ACEs.
1371
1372      ACE_POINTER = .NEW_ACE_HEAD[ACEQ_L_FLINK];
1373      UNTIL .ACE_POINTER EQL .NEW_ACE_HEAD[ACEQ_L_FLINK]
1374      DO
1375      BEGIN
1376      CH$MOVE (.S$BLOCK[ACE_POINTER[ACEQ_T_ACE], ACE$B_SIZE],
1377      ACE_POINTER[ACEQ_T_ACE], ACE);
1378      ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_ADDACLENT;
1379      ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACE$B_SIZE];
1380      ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1381      STATUS = $CHANGE_ACL (CHAN = .CHAN,
1382      OBJTYP = OBJECT_TYPE,
1383      OBJNAM = .OBJECT_NAME_DESC,
1384      ITMLST = ATR_ARGLIST,
1385      CONTXT = ACL_CONTEXT);
```

```
1385 1380 3 IF NOT .STATUS
1386 1381 3 THEN
1387 1382 4 BEGIN
1388 1383 4 SIGNAL (SET$ WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1389 1384 4 RETURN SET$ WRITEERR OR ST$M_INHIB_MSG;
1390 1385 3 END;
1391 1386 3 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1392 1387 2 END;
1393 1388 2
1394 1389 2 RETURN 1;
1395 1390 2
1396 1391 1 END;
```

! End of routine ADD_ACL

				03FC 00000	ADD_ACL: .WORD	Save R2,R3,R4,R5,R6,R7,R8,R9	1266
		59	00000000G	00 9E 00002	MOVAB	LIB\$SIGNAL, R9	
		58	00000000G	00 9E 00009	MOVAB	SYSS\$CHANGE_ACL, R8	
		57	0000' FCA8	CF 9E 00010	MOVAB	ACE, R7	
				C7 D4 00015	CLRL	ACL_CONTEXT	1306
48	F97C	C7		05 E1 00019	BBC	#5-FLAGS, 3\$	1310
	C8	A7	000600FF	8F D0 0001F	MOVL	#393471, ATR_ARGLIST	1314
	CC	A7		67 9E 00027	MOVAB	ACE, ATR_ARGLIST+4	1315
			FCA8	C7 9F 0002B	PUSHAB	ACL_CONTEXT	1320
				7E 7C 0002F	CLRQ	-(SP)	
			C8	A7 9F 00031	PUSHAB	ATR_ARGLIST	
			04	AC DD 00034	PUSHL	OBJECT_NAME_DESC	
			F988	C7 9F 00037	PUSHAB	OBJECT_TYPE	
			FCA4	C7 DD 0003B	PUSHL	CHAN	
		68		07 FB 0003F	CALLS	#7, SYSS\$CHANGE_ACL	
		56		50 D0 00042	MOVL	R0, STATUS	
		1F		56 E8 00045	BLBS	STATUS, 3\$	1321
				7E D4 00048	CLRL	-(SP)	1324
				56 DD 0004A	PUSHL	STATUS	
			04	AC DD 0004C	PUSHL	OBJECT_NAME_DESC	
			007710D4	8F DD 0004F	PUSHL	#7803092	
		69		04 FB 00055	CALLS	#4, LIB\$SIGNAL	
04	F980	C7	03	00 ED 00058	1\$: CMPZV	#0, #3, WORST_ERROR, #4	
				03 19 0005F	2\$: BLSS	2\$	
				00DE 31 00061	BRW	10\$	
				00D2 31 00064	2\$: BRW	9\$	
		6A	F97C	C7 E9 00067	3\$: BLBC	FLAGS, 7\$	1331
	0200	C7	0E0C	C7 D0 0006C	MOVL	OLD_ACE_HEAD, ACE_POINTER	1334
		50	0200	C7 D0 00073	MOVL	ACE_POINTER, R0	1335
		51	08	A0 9A 00078	MOVZBL	8(R0), R1	
67	08	A0		51 28 0007C	MOV3	R1, 8(R0), ACE	1336
	CA	A7		04 B0 00081	MOVW	#4, ATR_ARGLIST+2	1337
	C8	A7		67 9B 00085	MOVZBW	ACE, ATR_ARGLIST	1338
	CC	A7		67 9E 00089	MOVAB	ACE, ATR_ARGLIST+4	1339
			FCA8	C7 9F 0008D	PUSHAB	ACL_CONTEXT	1344
				7E 7C 00091	CLRQ	-(SP)	
			C8	A7 9F 00093	PUSHAB	ATR_ARGLIST	
			04	AC DD 00096	PUSHL	OBJECT_NAME_DESC	
			F988	C7 9F 00099	PUSHAB	OBJECT_TYPE	
			FCA4	C7 DD 0009D	PUSHL	CHAN	

68	07	FB	000A1	CALLS	#7, SYSSCHANGE_ACL	
56	50	D0	000A4	MOVL	R0, STATUS	
14	56	E8	000A7	BLBS	STATUS, 5\$	1345
	7E	D4	000AA	CLRL	-(SP)	1348
	56	DD	000AC	PUSHL	STATUS	
	04	AC	DD 000AE 4\$:	PUSHL	OBJECT_NAME_DESC	
	01	DD	000B1	PUSHL	#1	
	8F	DD	000B3	PUSHL	#7803092	
69	05	FB	000B9	CALLS	#5, LIB\$SIGNAL	
	9A	11	000BC	BRB	1\$	
	67	95	000BE 5\$:	TSTB	ACE	1351
	10	12	000C0	BNEQ	6\$	
09D0	8F	02	A7 B1 000C2	CMPW	ACE+2, #2512	1354
	08	13	000C8	BEQL	6\$	
	7E	D4	000CA	CLRL	-(SP)	1357
	02	A7	3C 000CC	MOVZWL	ACE+2, -(SP)	
		DC	11 000D0	BRB	4\$	
	FCA8	C7	D6 000D2 6\$:	INCL	ACL_CONTEXT	1361
0200	C7	OE14	C7 D0 000D6 7\$:	MOVL	NEW_ACE_HEAD, ACE_POINTER	1366
	50	0200	C7 D0 000DD 8\$:	MOVL	ACE_POINTER, R0	1367
	51	OE14	C7 9E 000E2	MOVAB	NEW_ACE_HEAD, R1	
	51		50 D1 000E7	CMPL	R0, R1	
			67 13 000EA	BEQL	12\$	
67	51	08	A0 9A 000EC	MOVZBL	8(R0), R1	1370
	08	A0	51 28 000F0	MOVCL	R1, 8(R0), ACE	1371
	CA	A7	01 B0 000F5	MOVW	#1, ATR_ARGLIST+2	1372
	C8	A7	67 9B 000F9	MOVZBW	ACE, ATR_ARGLIST	1373
	CC	A7	67 9E 000FD	MOVAB	ACE, ATR_ARGLIST+4	1374
		FCA8	C7 9F 00101	PUSHAB	ACL_CONTEXT	1379
			7E 7C 00105	CLRQ	-(SP)	
		C8	A7 9F 00107	PUSHAB	ATR_ARGLIST	
		04	AC DD 0010A	PUSHL	OBJECT_NAME_DESC	
		F988	C7 9F 0010D	PUSHAB	OBJECT_TYPE	
		FCA4	C7 DD 00111	PUSHL	CHAN	
	68		07 FB 00115	CALLS	#7, SYSSCHANGE_ACL	
	56		50 D0 00118	MOVL	R0, STATUS	
	2C		56 E8 0011B	BLBS	STATUS, 11\$	1380
			7E D4 0011E	CLRL	-(SP)	1383
		04	56 DD 00120	PUSHL	STATUS	
			AC DD 00122	PUSHL	OBJECT_NAME_DESC	
		01	DD 00125	PUSHL	#1	
		8F	DD 00127	PUSHL	#7803092	
	69	05	FB 0012D	CALLS	#5, LIB\$SIGNAL	
	03	00	ED C0130	CMPZV	#0, #3, WORST_ERROR, #4	
		09	18 00137	BGEQ	10\$	
04	F980	C7	8F D0 00139 9\$:	MOVL	#276238548, WORST_ERROR	
		50	8F D0 00142 10\$:	MOVL	#276238548, R0	1384
			04 00149	RET		
0200	C7	0200	D7 D0 0014A 11\$:	MOVL	ACE_POINTER, ACE_POINTER	1386
			8A 11 00151	BRB	8\$	1367
	50		01 D0 00153 12\$:	MOVL	#1, R0	1389
			04 00156	RET		1391

; Routine Size: 343 bytes, Routine Base: \$CODE\$ + 0F09


```
1398 1 ROUTINE DELETE_ACL (OBJECT_NAME_DESC) =
1399 1
1400 1 ++
1401 1
1402 1 FUNCTIONAL DESCRIPTION:
1403 1
1404 1     This routine deletes one or more ACEs (or the entire ACL) from
1405 1     the specified object.
1406 1
1407 1 CALLING SEQUENCE:
1408 1     ADD_ACL (ARG1)
1409 1
1410 1 INPUT PARAMETERS:
1411 1     ARG1: address of the FAB
1412 1
1413 1 IMPLICIT INPUTS:
1414 1     none
1415 1
1416 1 OUTPUT PARAMETERS:
1417 1     none
1418 1
1419 1 IMPLICIT OUTPUTS:
1420 1     none
1421 1
1422 1 ROUTINE VALUE:
1423 1     1 if successful
1424 1     error code otherwise
1425 1
1426 1 SIDE EFFECTS:
1427 1     none
1428 1
1429 1 --
1430 1
1431 2 BEGIN
1432 2
1433 2 LOCAL
1434 2     STATUS;
1435 2
1436 2     ! Local routine return status
1437 2
1438 2 ! If there were ACEs given on the /ACL qualifier, just those specified ACEs
1439 2 ! are deleted. Otherwise, the entire ACL is deleted.
1440 2
1441 2 IF .OLD_ACE_HEAD[ACEQ_L_FLINK] NEQA OLD_ACE_HEAD[ACEQ_L_FLINK]
1442 2 THEN
1443 2     BEGIN
1444 2
1445 2 ! Before deleting any of the given ACEs, make sure that they all exist.
1446 2
1447 2     ACE_POINTER = .OLD_ACE_HEAD[ACEQ_L_FLINK];
1448 2     UNTIL .ACE_POINTER EQA OLD_ACE_HEAD[ACEQ_L_FLINK]
1449 2     DO
1450 2         BEGIN
1451 2             CH$MOVE (.SBBLOCK[ACE_POINTER[ACEQ_T_ACE], ACESB_SIZE],
1452 2                     ACE_POINTER[ACEQ_T_ACE], ACE);
1453 2             ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_FNDACLENT;
1454 2             ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACESB_SIZE];
1455 2             ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1456 2             STATUS = $CHANGE_ACL (CHAN = .CHAN,
```

```
1455 P 1449 4 OBJTYP = OBJECT TYPE
1456 P 1450 4 OBJNAM = .OBJECT NAME_DESC,
1457 P 1451 4 ITMLST = ATR_ARGLIST,
1458 1452 4 CONTXT = ACL_CONTEXT);
1459 1453 4 IF NOT .STATUS
1460 1454 4 THEN
1461 1455 5 BEGIN
1462 1456 5 IF .STATUS NEQ SSS_ACLEMPTY
1463 1457 5 AND .STATUS NEQ SSS_NOENTRY
1464 1458 5 THEN
1465 1459 6 BEGIN
1466 1460 6 SIGNAL (SETS_WRITEERR, 1, .OBJECT NAME_DESC, .STATUS, 0);
1467 1461 6 RETURN SETS_WRITEERR OR ST$M_INHIB_MSG;
1468 1462 5 END;
1469 1463 5 ACE_DESC[DSC$W_LENGTH] = .SBBLOCK[ACE_POINTER[ACEQ_T_ACE], ACE$B_SIZE];
1470 1464 5 ACE_DESC[DSC$A_POINTER] = ACE_POINTER[ACEQ_T_ACE];
1471 1465 5 ACE_TEXT_DESC[DSC$W_LENGTH] = 3072;
1472 1466 5 ACE_TEXT_DESC[DSC$A_POINTER] = ACE_TEXT;
1473 P 1467 5 $FORMAT_ACL (ACLENT = ACE_DESC,
1474 P 1468 5 ACLEN = ACE_TEXT_DESC[DSC$W_LENGTH],
1475 P 1469 5 ACLSTR = ACE_TEXT_DESC,
1476 P 1470 5 WIDTH = %REF(80),
1477 P 1471 5 TRMDSC = $DESCRIPTOR (%CHAR(13), %CHAR(10)),
1478 1472 5 INDENT = %REF(4));
1479 1473 5 SIGNAL (SETS_NOSUCHACE, 2, .OBJECT NAME_DESC, ACE_TEXT_DESC);
1480 1474 4 END;
1481 1475 4 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1482 1476 3 END;
1483 1477 3
1484 1478 3 ! Delete the specified ACEs.
1485 1479 3
1486 1480 3 ACE_POINTER = .OLD_ACE_HEAD[ACEQ_L_FLINK];
1487 1481 3 UNTIL .ACE_POINTER = OLD_ACE_HEAD[ACEQ_L_FLINK]
1488 1482 3 DO
1489 1483 4 BEGIN
1490 1484 4 CH$MOVE (.SBBLOCK[ACE_POINTER[ACEQ_T_ACE], ACE$B_SIZE],
1491 1485 4 ACE_POINTER[ACEQ_T_ACE], ACE);
1492 1486 4 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_DELACLENT;
1493 1487 4 ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACE$B_SIZE];
1494 1488 4 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1495 P 1489 4 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1496 P 1490 4 OBJTYP = OBJECT TYPE,
1497 P 1491 4 OBJNAM = .OBJECT NAME_DESC,
1498 P 1492 4 ITMLST = ATR_ARGLIST,
1499 1493 4 CONTXT = ACL_CONTEXT);
1500 1494 4 IF NOT .STATUS
1501 1495 4 THEN
1502 1496 5 BEGIN
1503 1497 5 SIGNAL (SETS_WRITEERR, 1, .OBJECT NAME_DESC, .STATUS, 0);
1504 1498 5 RETURN SETS_WRITEERR OR ST$M_INHIB_MSG;
1505 1499 5 END;
1506 1500 4 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1507 1501 3 END;
1508 1502 3 END
1509 1503 2 ELSE
1510 1504 3 BEGIN
1511 1505 3
```

```
1512 1506 3 ! Delete any ACL that currently exists on the object.
1513 1507
1514 1508 ATR_ARGLIST[0, ITMSW_ITMCO] = ACL$C_DELETEACL;
1515 1509 ATR_ARGLIST[0, ITMSW_BUF$IZ] = ACL$S_DELETEACL;
1516 1510 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1517 1511 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1518 1512 OBJTYP = OBJECT TYPE,
1519 1513 OBJNAM = .OBJECT_NAME_DESC,
1520 1514 ITMLST = ATR_ARGLIST,
1521 1515 CONXT = ACL_CONTEXT);
1522 1516 IF NOT .STATUS
1523 1517 THEN
1524 1518 BEGIN
1525 1519 SIGNAL (SET$ WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1526 1520 RETURN SET$ WRITEERR OR ST$M_INHTB_MSG;
1527 1521 END;
1528 1522 END;
1529 1523
1530 1524 RETURN 1;
1531 1525
1532 1526 1 END; ! End of routine DELETE_ACL
```

.PSECT \$PLITS, NOWRT, NOEXE, 2

```
0D 00354 P.ACX: .ASCII <13>
0A 00355 .ASCII <10>
00000002 00356 .BLKB 2
00000000 00358 P.ACW: .LONG 2
00000000 0035C .ADDRESS P.ACX
```

.EXTRN SYS\$FORMAT_ACL

.PSECT \$CODE\$, NOWRT, 2

OFFC 0000 DELETE_ACL:

5B	00000000G	00	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1392
5A	00000000G	00	9E	00009	MOVAB	SET\$ NOSUCHACE, R11	
59	00000000G	00	9E	00010	MOVAB	LIB\$SIGNAL, R10	
58	00000000G	00	9E	00017	MOVAB	SYSSCHANGE_ACL, R9	
5E	00000000G	CF	9E	00017	MOVAB	ACE_POINTER, R8	
56	04	08	C2	0001C	SUBL2	#8, SP	
50	0C0C	C8	D0	0001F	MOVL	OBJECT_NAME_DESC, R6	1452
50	0C0C	C8	D1	00023	MOVAB	OLD_ACE_HEAD, R0	1433
			D1	00028	CMPL	OLD_ACE_HEAD, R0	
		03	12	0002D	BNEQ	1\$	
		014D	31	0002F	BRW	13\$	
68	0C0C	C8	D0	00032	MOVL	OLD_ACE_HEAD, ACE_POINTER	1439
50		68	D0	00037	MOVL	ACE_POINTER, R0	1440
51	0C0C	C8	9E	0003A	MOVAB	OLD_ACE_HEAD, R1	
51		50	D1	0003F	CMPL	R0, R1	
		03	12	00042	BNEQ	3\$	
		00E0	31	00044	BRW	9\$	
		A0	9A	00047	MOVZBL	8(R0), R1	1443
FE00	C8	08	51	28	MOVCS	R1, 8(R0), ACE	1444
FDCA	C8	04	B0	00052	MOVW	#4, ATR_ARGLIST+2	1445

	FDC8	C8	FE00	C8	9B	00057	MOVZBW	ACE, ATR_ARGLIST	1446	
	FDCC	C8	FE00	C8	9E	0005E	MOVAB	ACE, ATR_ARGLIST+4	1447	
			FAAB	C8	9F	00065	PUSHAB	ACL_CONTEXT	1452	
				7E	7C	00069	CLRQ	-(SP)		
			FDC8	C8	9F	0006B	PUSHAB	ATR_ARGLIST		
				56	DD	0006F	PUSHL	R6		
			F788	C8	9F	00071	PUSHAB	OBJECT_TYPE		
			FAA4	C8	DD	00075	PUSHL	CHAN		
		69		07	FB	00079	CALLS	#7, SYSSCHANGE_ACL		
		57		50	DD	0007C	MOVL	R0, STATUS		
		03		57	E9	0007F	BLBC	STATUS, 4\$	1453	
				009C	31	00082	BRW	8\$		
	000009D0	8F		57	D1	00085	4\$:	CMPL	STATUS, #2512	1456
				28	13	0008C	BEQL	7\$		
	000009D8	8F		57	D1	0008E	CMPL	STATUS, #2520	1457	
				1F	13	00095	BEQL	7\$		
				7E	D4	00097	5\$:	CLRL	-(SP)	1460
		7E		56	7D	00099	MOVQ	R6, -(SP)		
				01	DD	0009C	PUSHL	#1		
			007710D4	8F	DD	0009E	PUSHL	#7803092		
		6A		05	FB	000A4	CALLS	#5, LIBSSIGNAL		
04	F780	C8		00	ED	000A7	CMPZV	#0, #3, WORST_ERROR, #4		
		03		03	19	000AE	BLSS	6\$		
				011B	31	000B0	BRW	15\$		
				010F	31	000B3	6\$:	BRW	14\$	
				68	DD	000B6	7\$:	MOVL	ACE_POINTER, R0	1463
	FDF8	C8	08	A0	9B	000B9	MOVZBW	8(R0), ACE_DESC		
	FDFC	C8	08	A0	9E	000BF	MOVAB	8(R0), ACE_DESC+4	1464	
	04	A8	0C00	8F	80	000C5	MOVW	#3072, ACE_TEXT_DESC	1465	
	08	A8	0C	A8	9E	000CB	MOVAB	ACE_TEXT, ACE_TEXT_DESC+4	1466	
				7E	D4	000D0	CLRL	-(SP)	1472	
	08	AE		04	DD	000D2	MOVL	#4, 8(SP)		
			08	AE	9F	000D6	PUSHAB	8(SP)		
			0000*	CF	9F	000D9	PUSHAB	P.ACW		
	0C	AE		50	8F	9A	000DD	MOVZBL	#80, 12(SP)	
				0C	AE	9F	000E2	PUSHAB	12(SP)	
				04	A8	9F	000E5	PUSHAB	ACE_TEXT_DESC	
				04	A8	9F	000E8	PUSHAB	ACE_TEXT_DESC	
			FDF8	C8	9F	000EB	PUSHAB	ACE_DESC		
00000000G	00			07	FB	000EF	CALLS	#7, SYSSFORMAT_ACL		
			04	A8	9F	000F6	PUSHAB	ACE_TEXT_DESC	1473	
				56	DD	000F9	PUSHL	R6		
				02	DD	000FB	PUSHL	#2		
				5B	DD	000FD	PUSHL	R11		
				04	FB	000FF	CALLS	#4, LIBSSIGNAL		
	6A			6B	9E	00102	MOVAB	SET\$ NOSUCHACE, R0		
	50			50	E8	00105	BLBS	R0, 8\$		
	19			00	9E	00108	MOVAB	<SET\$ NOSUCHACE&7>, R0		
50	F780	C8	00000000*	00	ED	0010F	CMPZV	#0, #3, WORST_ERROR, R0		
				09	18	00116	BGEQ	8\$		
				00	9E	00118	MOVAB	<SET\$ NOSUCHACE!268435456>, WORST_ERROR	1475	
	F780	C8	00000000*	00	9E	00118	MOVAB	ACE_POINTER, ACE_POINTER	1480	
		78		98	DD	00121	8\$:	MOVL	2\$	
				FF10	31	00124	BRW	2\$		
		68	0C0C	C8	DD	00127	9\$:	MOVL	OLD_ACE_HEAD, ACE_POINTER	1481
		50		68	DD	0012C	10\$:	MOVL	ACE_POINTER, R0	
		51	0C0C	C8	9E	0012F	MOVAB	OLD_ACE_HEAD, R1		
		51		50	D1	00134	CMPL	R0, -R1		

					03	12	00137		BNEQ	11\$		
					009A	31	00139		BRW	16\$		
FE00	C8	08	51	08	A0	9A	0013C	11\$:	MOVZBL	8(R0), R1	1484	
		FDCA	C8		51	28	00140		MOVCS	R1, 8(R0), ACE	1485	
		FDC8	C8	FE00	02	B0	00147		MOVW	#2, ATR_ARGLIST+2	1486	
		FDCC	C8	FE00	C8	9B	0014C		MOVZBW	ACE, ATR_ARGLIST	1487	
				FAA8	C8	9E	00153		MOVAB	ACE, ATR_ARGLIST+4	1488	
					7E	7C	0015A		PUSHAB	ACL_CONTEXT	1493	
				FDC8	C8	9F	0015E		CLRQ	-(SP)		
					56	DD	00160		PUSHAB	ATR_ARGLIST		
				F788	C8	9F	00166		PUSHL	R6		
				FAA4	C8	DD	0016A		PUSHAB	OBJECT_TYPE		
					07	FB	0016E		PUSHL	CHAN		
			69		50	DD	00171		CALLS	#7, SYSS\$CHANGE_ACL		
			57		57	EB	00174		MOVL	R0, STATUS		
			03		FF1D	31	00177		BLBS	STATUS, 12\$	1494	
					98	DD	0017A	12\$:	BRW	5\$		
			78		AD	11	0017D		MOVL	ACE_POINTER, ACE_POINTER	1500	
					8F	DD	0017F	13\$:	BRB	10\$	1481	
		FDC8	C8	000600FF	C8	9E	00188		MOVL	#393471, ATR_ARGLIST	1509	
		FDCC	C8	FE00	C8	9F	0018F		MOVAB	ACE, ATR_ARGLIST+4	1510	
				FAA8	C8	7E	7C	00193	PUSHAB	ACL_CONTEXT	1515	
					7E	7C	00193		CLRQ	-(SP)		
				FDC8	C8	9F	00195		PUSHAB	ATR_ARGLIST		
					56	DD	00199		PUSHL	R6		
				F788	C8	9F	0019B		PUSHAB	OBJECT_TYPE		
				FAA4	C8	DD	0019F		PUSHL	CHAN		
					07	FB	001A3		CALLS	#7, SYSS\$CHANGE_ACL		
			69		50	DD	001A6		MOVL	R0, STATUS		
			57		57	EB	001A9		BLBS	STATUS, 16\$	1516	
			2A		7E	D4	001AC		CLRL	-(SP)	1519	
					56	7D	001AE		MOVQ	R6, -(SP)		
			7E		01	DD	001B1		PUSHL	#1		
				007710D4	8F	DD	001B3		PUSHL	#7803092		
					05	FB	001B9		CALLS	#5, LIB\$SIGNAL		
04	F780	C8	6A		00	ED	001BC		CMPZV	#0, #3, WORST_ERROR, #4		
			03		09	18	001C3		BGEQ	15\$		
					8F	DD	001C5	14\$:	MOVL	#276238548, WORST_ERROR	1520	
		F780	C8	107710D4	8F	DD	001CE	15\$:	MOVL	#276238548, R0		
			50	107710D4	04	001D5			RET		1524	
					01	DD	001D6	16\$:	MOVL	#1, R0	1526	
			50		04	001D9			RET			

; Routine Size: 474 bytes, Routine Base: \$CODE\$ + 1060

```
1534 1 ROUTINE REPLACE_ACL (OBJECT_NAME_DESC) =
1535 1
1536 1 ++
1537 1
1538 1 FUNCTIONAL DESCRIPTION:
1539 1
1540 1 This routine deletes the indicated ACEs, and then replaces them
1541 1 with the new ones specified on the /REPLACE qualifier.
1542 1
1543 1 CALLING SEQUENCE:
1544 1 ADD_ACL (ARG1)
1545 1
1546 1 INPUT PARAMETERS:
1547 1 ARG1: address of the FAB
1548 1
1549 1 IMPLICIT INPUTS:
1550 1 none
1551 1
1552 1 OUTPUT PARAMETERS:
1553 1 none
1554 1
1555 1 IMPLICIT OUTPUTS:
1556 1 none
1557 1
1558 1 ROUTINE VALUE:
1559 1 1 if successful
1560 1 error code otherwise
1561 1
1562 1 SIDE EFFECTS:
1563 1 none
1564 1
1565 1 --
1566 1
1567 2 BEGIN
1568 2
1569 2 LOCAL
1570 2 OLD_ACLCTX, ! Old ACL context
1571 2 STATUS; ! Local routine return status
1572 2
1573 2 ! Before deleting any of the given ACEs, make sure that they all exist and
1574 2 ! the order is correct.
1575 2
1576 2 OLD_ACLCTX = 0;
1577 2 ACE_POINTER = .OLD_ACE_HEAD[ACEQ_L_FLINK];
1578 2 UNTIL .ACE_POINTER=EQLA OLD_ACE_HEAD[ACEQ_L_FLINK]
1579 2 DO
1580 2 BEGIN
1581 2 CH$MOVE (.SBBLOCK[ACE_POINTER[ACEQ_T_ACE], ACESB_SIZE],
1582 2 ACE_POINTER[ACEQ_T_ACE], ACE);
1583 2 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_FNDACLENT;
1584 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACESB_SIZE];
1585 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1586 2 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1587 2 OBJTYP = OBJECT_TYPE,
1588 2 OBJNAM = .OBJECT_NAME_DESC,
1589 2 ITMLST = ATR_ARGLIST,
1590 2 CONTXT = ACL_CONTEXT);
```

```
1591 1584 3 IF NOT .STATUS
1592 1585 3 THEN
1593 1586 4 BEGIN
1594 1587 4 IF .STATUS NEQ $$$_ACLEMPY
1595 1588 4 AND .STATUS NEQ $$$_NOENTRY
1596 1589 4 THEN
1597 1590 5 BEGIN
1598 1591 5 SIGNAL (SETS_WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1599 1592 5 RETURN SETS_WRITEERR OR $$$_INHIB_MSG;
1600 1593 4 END;
1601 1594 4 ACE_DESC[DSCSW_LENGTH] = .$$$BLOCK[ACE_POINTER[ACEQ_T_ACE], ACESB_SIZE];
1602 1595 4 ACE_DESC[DSCSA_POINTER] = ACE_POINTER[ACEQ_T_ACE];
1603 1596 4 ACE_TEXT_DESC[DSCSW_LENGTH] = 3072;
1604 1597 4 ACE_TEXT_DESC[DSCSA_POINTER] = ACE_TEXT;
1605 1598 4 $FORMAT_ACL (ACLENT = ACE_DESC,
1606 1599 4 ACLEN = ACE_TEXT_DESC[DSCSW_LENGTH],
1607 1600 4 ACLSTR = ACE_TEXT_DESC,
1608 1601 4 WIDTH = %REF(80),
1609 1602 4 TRMDSC = $DESCRIPTOR (%CHAR(13), %CHAR(10)),
1610 1603 4 INDENT = %REF(4));
1611 1604 4 SIGNAL (SETS_NOSUCHACE, 2, .OBJECT_NAME_DESC, ACE_TEXT_DESC);
1612 1605 4 RETURN SETS_NOSUCHACE OR $$$_INHIB_MSG;
1613 1606 3 END;
1614 1607 3
1615 1608 3 ! The ACE exists. Is the ordering correct?
1616 1609 3
1617 1610 3 IF .OLD_ACLCTX NEQ 0
1618 1611 3 THEN
1619 1612 4 BEGIN
1620 1613 4 IF .OLD_ACLCTX<0,24> + 1 NEQ .ACL_CONTEXT
1621 1614 4 THEN
1622 1615 5 BEGIN
1623 1616 5 SIGNAL (SETS_IVORDER, 1, .OBJECT_NAME_DESC);
1624 1617 5 RETURN SETS_IVORDER OR $$$_INHIB_MSG;
1625 1618 4 END;
1626 1619 3 END;
1627 1620 3 OLD_ACLCTX = .ACL_CONTEXT;
1628 1621 3 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1629 1622 3 END;
1630 1623 3
1631 1624 2 ! Delete any ACEs specified on the /ACL qualifier.
1632 1625 2
1633 1626 2 ACE_POINTER = .OLD_ACE_HEAD[ACEQ_L_FLINK];
1634 1627 2 UNTIL .ACE_POINTER=EQLA .OLD_ACE_HEAD[ACEQ_L_FLINK]
1635 1628 2 DO
1636 1629 3 BEGIN
1637 1630 3 CHSMOVE (.$$$BLOCK[ACE_POINTER[ACEQ_T_ACE], ACESB_SIZE],
1638 1631 3 ACE_POINTER[ACEQ_T_ACE], ACE);
1639 1632 3 ATR_ARGLIST[0, ITMSW_ITMCD] = ACLSC_DELACLENT;
1640 1633 3 ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACESB_SIZE];
1641 1634 3 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1642 1635 3 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1643 1636 3 OBJTYP = .OBJECT_TYPE,
1644 1637 3 OBJNAM = .OBJECT_NAME_DESC,
1645 1638 3 ITMLST = ATR_ARGLIST,
1646 1639 3 CONTXT = ACL_CONTEXT);
1647 1640 3 IF NOT .STATUS
```

```
1648 1641 3 THEN
1649 1642 4 BEGIN
1650 1643 4 SIGNAL (SETS_WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1651 1644 4 RETURN SETS_WRITEERR OR $TSM_INHIB_MSG;
1652 1645 4 END;
1653 1646 3 IF .ACE[ACESB_SIZE] EQL 0
1654 1647 3 THEN
1655 1648 4 BEGIN
1656 1649 4 IF .ACE[ACESW_FLAGS] EQL $$$_ACLEMPY
1657 1650 4 THEN EXITLOOP
1658 1651 4 ELSE
1659 1652 5 BEGIN
1660 1653 5 SIGNAL (SETS_WRITEERR, 1, .OBJECT_NAME_DESC, .ACE[ACESW_FLAGS], 0);
1661 1654 5 RETURN SETS_WRITEERR OR $TSM_INHIB_MSG;
1662 1655 5 END;
1663 1656 4 END;
1664 1657 3 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1665 1658 2 END;
1666 1659 2
1667 1660 2 ! Add the new ACEs specified on the /REPLACE qualifier.
1668 1661 2
1669 1662 2 ACE_POINTER = .NEW_ACE_HEAD[ACEQ_L_FLINK];
1670 1663 2 UNTIL .ACE_POINTER EQLA NEW_ACE_HEAD[ACEQ_L_FLINK]
1671 1664 2 DO
1672 1665 3 BEGIN
1673 1666 3 CHSMOVE (.SBBLOCK[ACE_POINTER[ACEQ_T_ACE], ACESB_SIZE],
1674 1667 3 ACE_POINTER[ACEQ_T_ACE], ACE);
1675 1668 3 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_ADDACLENT;
1676 1669 3 ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACESB_SIZE];
1677 1670 3 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1678 1671 3 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1679 1672 3 OBJTYP = OBJECT_TYPE,
1680 1673 3 OBJNAM = .OBJECT_NAME_DESC,
1681 1674 3 ITMLST = ATR_ARGLIST,
1682 1675 3 CONTXT = ACL_CONTEXT);
1683 1676 3
1684 1677 3 IF NOT .STATUS
1685 1678 4 THEN
1686 1679 4 BEGIN
1687 1680 4 SIGNAL (SETS_WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
1688 1681 4 RETURN SETS_WRITEERR OR $TSM_INHIB_MSG;
1689 1682 3 END;
1690 1683 2 ACE_POINTER = .ACE_POINTER[ACEQ_L_FLINK];
1691 1684 2 END;
1692 1685 2 RETURN 1;
1693 1686 2
1694 1687 1 END;
```

! End of routine REPLACE_ACL

```
.PSECT $SPLITS,NOWRT,NOEXE,2
OD 00360 P.ACZ: .ASCII <13>
OA 00361 .ASCII <10>
00362 .BLKB 2
00000002 00364 P.ACY: .LONG 2
00000000 00368 .ADDRESS P.ACZ
```



```
.PSECT $CODE$,NOWRT,2

OFFC 00000 REPLACE_ACL:
5B 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 : 1527
5A 00000000G 00 9E 00009 MOVAB SY$CHANGE_ACL, R11
59 0000' CF 9E 00010 MOVAB LIB$SIGNAL, R10
5E 08 C2 00015 MOVAB ACE_POINTER, R9
58 D4 00018 SUBL2 #8, -SP
69 0C0C C9 D0 0001A CLRL OLD_ACLCTX : 1569
56 04 AC D0 0001F MOVL OLD_ACE_HEAD, ACE_POINTER : 1570
50 69 D0 00023 1$: MOVL OBJECT_NAME_DESC, R6 : 1583
51 0C0C C9 9E 00026 MOVL ACE_POINTER, R0 : 1571
51 50 D1 0002B MOVAB OLD_ACE_HEAD, R1
03 12 0002E CMPL R0, R1
0140 31 00030 BNEQ 2$
51 08 A0 9A 00033 2$: MOVZBL 8(R0), R1 : 1574
A0 C9 51 28 00037 MOVCL R1, 8(R0), ACE : 1575
FDC8 C9 04 B0 0003E MOVW #4, ATR_ARGLIST+2 : 1576
FDC8 C9 FE00 C9 9B 00043 MOVZBW ACE, ATR_ARGLIST : 1577
FE00 C9 FE00 C9 9E 0004A MOVAB ACE, ATR_ARGLIST+4 : 1578
FAA8 C9 9F 00051 PUSHAB ACL_CONTEXT : 1583
7E 7C 00055 CLRL -(SP)
FDC8 C9 9F 00057 PUSHAB ATR_ARGLIST
56 DD 0005B PUSHL R6
F788 C9 9F 0005D PUSHAB OBJECT_TYPE
FAA4 C9 DD 00061 PUSHL CHAN
6B 07 FB 00065 CALLS #7, SY$CHANGE_ACL
57 50 D0 00068 MOVL R0, STATUS
03 57 E9 0006B BLBC STATUS, 3$ : 1584
00AD 31 0006E BRW 9$ : 1587
000009D0 8F 57 D1 00071 3$: CMPL STATUS, #2512 : 1588
000009D8 8F 29 13 00078 BEQL 7$ : 1591
57 D1 0007A CMPL STATUS, #2520 : 1594
20 13 00081 BEQL 7$
7E D4 00083 4$: CLRL -(SP) : 1595
57 DD 00085 5$: PUSHL STATUS : 1596
56 DD 00087 PUSHL R6 : 1597
01 DD 00089 PUSHL #1 : 1603
007710D4 8F DD 0008B PUSHL #7803092
6A 05 FB 00091 CALLS #5, LIB$SIGNAL
03 00 ED 00094 CMPZV #0, #3, WORST_ERROR, #4
03 19 0009B BLSS 6$
01B0 31 0009D BRW 19$
01A4 31 000A0 6$: BRW 18$ : 1594
50 C9 08 A0 9B 000A6 MOVL ACE_POINTER, R0 : 1595
FDFB C9 08 A0 9E 000AC MOVZBW 8(R0), ACE_DESC : 1596
FDFC C9 08 A0 9E 000AC MOVAB 8(R0), ACE_DESC+4 : 1597
04 A9 0C00 8F B0 000B2 MOVW #3072, ACE_TEXT_DESC : 1603
08 A9 0C A9 9E 000B8 MOVAB ACE_TEXT, ACE_TEXT_DESC+4
7E D4 000BD CLRL -(SP)
08 AE 04 D0 000BF MOVL #4, 8(SP)
08 AE 9F 000C3 PUSHAB 8(SP)
0000' CF 9F 000C6 PUSHAB P.ACY
```

	OC	AE	50	8F	9A	000CA	MOVZBL	#80, 12(SP)		
			OC	AE	9F	000CF	PUSHAB	12(SP)		
			04	A9	9F	000D2	PUSHAB	ACE_TEXT_DESC		
			04	A9	9F	000D5	PUSHAB	ACE_TEXT_DESC		
			FDF8	C9	9F	000D8	PUSHAB	ACE_DESC		
	00000000G	00		07	FB	000DC	CALLS	#7, SYSS\$FORMAT_ACL		
		04		A9	9F	000E3	PUSHAB	ACE_TEXT_DESC		1604
				56	DD	000E6	PUSHL	R6		
				02	DD	000E8	PUSHL	#2		
		00000000G		00	9F	000EA	PUSHAB	SET\$ NOSUCHACE		
	6A			04	FB	000F0	CALLS	#4, [IB\$SIGNAL		
	50	00000000G		00	9E	000F3	MOVAB	SET\$ NOSUCHACE, R0		
	19			50	E8	000FA	BLBS	R0, 8\$		
50	F780	C9	50	00000000*	00	9E	000FD	MOVAB	<SET\$ NOSUCHACE87>, R0	
			03	00	ED	00104	CMPZV	#0, #3, WORST_ERROR, R0		
				09	18	0010B	BGEQ	8\$		
	F780	C9	00000000*	00	9E	0010D	MOVAB	<SET\$ NOSUCHACE!268435456>, WORST_ERROR		
		50	00000000*	00	9E	00116	MOVAB	<SET\$ NOSUCHACE!268435456>, R0		1605
				04	00	0011D	RET			
				58	D5	0011E	TSTL	OLD_ACLCTX		1610
				46	13	00120	BEQL	11\$		
50		58	18	00	EF	00122	EXTZV	#0, #24, OLD_ACLCTX, R0		1613
				50	D6	00127	INCL	R0		
	FAA8	C9	50	D1	00129	CMP	R0, ACL_CONTEXT			
				38	13	0012E	BEQL	11\$		
				56	DD	00130	PUSHL	R6		1616
				01	DD	00132	PUSHL	#1		
		00000000G		00	9F	00134	PUSHAB	SET\$ IVORDER		
	6A			03	FB	0013A	CALLS	#3, [IB\$SIGNAL		
	50	00000000G		00	9E	0013D	MOVAB	SET\$ IVORDER, R0		
	19			50	E8	00144	BLBS	R0, 10\$		
50	F780	C9	50	00000000*	00	9E	00147	MOVAB	<SET\$ IVORDER87>, R0	
			03	00	ED	0014E	CMPZV	#0, #3, WORST_ERROR, R0		
				09	18	00155	BGEQ	10\$		
	F780	C9	00000000*	00	9E	00157	MOVAB	<SET\$ IVORDER!268435456>, WORST_ERROR		
		50	00000000*	00	9E	00160	MOVAB	<SET\$ IVORDER!268435456>, R0		1617
				04	00	00167	RET			
		58	FAA8	C9	D0	00168	MOVL	ACL_CONTEXT, OLD_ACLCTX		1620
		79		99	D0	0016D	MOVL	ACE_POINTER, ACE_POINTER		1621
				FE00	31	00170	BRW	1\$		1571
		69	OCOC	C9	D0	00173	MOVL	OLD_ACE_HEAD, ACE_POINTER		1626
		50		69	D0	00178	MOVL	ACE_POINTER, R0		1627
		51	OCOC	C9	9E	0017B	MOVAB	OLD_ACE_HEAD, R1		
		51		50	D1	00180	CMP	R0, R1		
				5C	13	00183	BEQL	16\$		
		51	08	A0	9A	00185	MOVZBL	8(R0), R1		1630
	FE00	C9	A0	51	28	00189	MOV3	R1, 8(R0), ACE		1631
		FDCA	C9	02	B0	00190	MOVW	#2, ATR_ARGLIST+2		1632
		FDC8	C9	FE00	C9	9B	MOVZBW	ACE, ATR_ARGLIST		1633
		FDC8	C9	FE00	C9	9E	MOVAB	ACE, ATR_ARGLIST+4		1634
				FAA8	C9	9F	PUSHAB	ACL_CONTEXT		1639
					7E	7C	CLRQ	-(SP)		
				FDC8	C9	9F	PUSHAB	ATR_ARGLIST		
					56	DD	PUSHL	R6		
				F788	C9	9F	PUSHAB	OBJECT_TYPE		
				FAA4	C9	DD	PUSHL	CHAN		
			6B		07	FB	CALLS	#7, SYSS\$CHANGE_ACL		

57	50	D0	0018A	MOVL	R0, STATUS	
03	57	E8	0018D	BLBS	STATUS, 14\$	1640
	FEC0	31	001C0	BRW	4\$	
	FE00	C9	95 001C3	TSTB	ACE	1646
		13	12 001C7	BNEQ	15\$	
09D0	8F	FE02	C9 B1 001C9	CMPW	ACE+2, #2512	1649
		0F	13 001D0	BEQL	16\$	
	7E	FE02	7E D4 001D2	CLRL	-(SP)	1653
		C9	3C 001D4	MOVZWL	ACE+2, -(SP)	
	FEAB	31	001D9	BRW	5\$	
	99	D0	001DC	MOVL	ACE_POINTER, ACE_POINTER	1657
	97	11	001DF	BRB	13\$	1627
	69	OC14	C9 D0 001E1	MOVL	NEW_ACE_HEAD, ACE_POINTER	1662
	50		69 D0 001E6	MOVL	ACE_POINTER, R0	1663
	51	OC14	C9 9E 001E9	MOVAB	NEW_ACE_HEAD, R1	
	51		50 D1 001EE	CMP	R0, R1	
		6A	13 001F1	BEQL	21\$	
	FE0D	C9	A0 9A 001F3	MOVZBL	8(R0), R1	1666
		08	51 28 001F7	MOVCL	R1, 8(R0), ACE	1667
	FDCA	C9	01 B0 001FE	MOVW	#1, ATR_ARGLIST+2	1668
	FDC8	C9	FE00 C9 9B 00203	MOVZBW	ACE, ATR_ARGLIST	1669
	FDC	C9	FE00 C9 9E 0020A	MOVAB	ACE, ATR_ARGLIST+4	1670
			FAA8 C9 9F 00211	PUSHAB	ACL_CONTEXT	1675
			7E 7C 00215	CLRL	-(SP)	
		FDC8	C9 9F 00217	PUSHAB	ATR_ARGLIST	
			56 DD 0021B	PUSHL	R6	
		F788	C9 9F 0021D	PUSHAB	OBJECT_TYPE	
		FAA4	C9 DD 00221	PUSHL	CHAN	
	6B		07 FB 00225	CALLS	#7, SYSSCHANGE_ACL	
	57		50 D0 00228	MOVL	R0, STATUS	
	2A		57 E8 0022B	BLBS	STATUS, 20\$	1676
			7E D4 0022E	CLRL	-(SP)	1679
			56 7D 00230	MOVQ	R6, -(SP)	
			01 DD 00233	PUSHL	#1	
		007710D4	8F DD 00235	PUSHL	#7803092	
		6A	05 FB 0023B	CALLS	#5, LIB\$SIGNAL	
		03	00 ED 0023E	CMPZV	#0, #3, WORST_ERROR, #4	
			09 18 00245	BGEQ	19\$	
			8F D0 00247	MOVL	#276238548, WORST_ERROR	
			8F D0 00250	MOVL	#276238548, R0	1680
			04 00257	RET		
		79	99 D0 00258	MOVL	ACE_POINTER, ACE_POINTER	1682
			89 11 0025B	BRB	17\$	1663
			01 D0 0025D	MOVL	#1, R0	1685
			04 00260	RET		1687

; Routine Size: 609 bytes, Routine Base: \$CODE\$ + 123A

```
1696 1688 1 ROUTINE COPY_ACL (OBJECT_NAME_DESC) =
1697 1689 1
1698 1690 1 ++
1699 1691 1
1700 1692 1 FUNCTIONAL DESCRIPTION:
1701 1693 1
1702 1694 1 This routine is called to copy the ACL from the specified input object
1703 1695 1 to the selected output object. It is also used to delete the ACL of
1704 1696 1 a object.
1705 1697 1
1706 1698 1 CALLING SEQUENCE:
1707 1699 1 COPY_ACL (ARG1)
1708 1700 1
1709 1701 1 INPUT PARAMETERS:
1710 1702 1 ARG1: address of the FAB
1711 1703 1
1712 1704 1 IMPLICIT INPUTS:
1713 1705 1 none
1714 1706 1
1715 1707 1 OUTPUT PARAMETERS:
1716 1708 1 none
1717 1709 1
1718 1710 1 IMPLICIT OUTPUTS:
1719 1711 1 none
1720 1712 1
1721 1713 1 ROUTINE VALUE:
1722 1714 1 1 if successful
1723 1715 1 error code otherwise
1724 1716 1
1725 1717 1 SIDE EFFECTS:
1726 1718 1 The ACL is copied from one object to another.
1727 1719 1
1728 1720 1 --
1729 1721 1
1730 1722 2 BEGIN
1731 1723 2
1732 1724 2 LOCAL
1733 1725 2 DEVICE_DESC : $BLOCK [DSC$C_S_BLN], ! Device name descr
1734 1726 2 DEVICE : $BLOCK [NAM$C_DVI], ! Device name storage
1735 1727 2 OBJECT_FIB_DESC : $BLOCK [DSC$C_S_BLN], ! Object's FIB descr
1736 1728 2 OBJECT_FIB : $BLOCK [FIB$C_LENGTH], ! Object's FIB
1737 1729 2 STATUS; ! Local routine return status
1738 1730 2
1739 1731 2 ! Delete any ACL that currently exists on the object.
1740 1732 2
1741 1733 2 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_DELETEACL;
1742 1734 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_DELETEACL;
1743 1735 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1744 1736 2 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1745 1737 2 OBJTYP = OBJECT TYPE,
1746 1738 2 OBJNAM = .OBJECT_NAME_DESC,
1747 1739 2 ITMLST = ATR_ARGLIST,
1748 1740 2 CONTXT = ACL_CONTEXT);
1749 1741 2 IF NOT .STATUS
1750 1742 2 THEN
1751 1743 2 BEGIN
1752 1744 2 SIGNAL (SET$WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
```



```
1753 1745 3 RETURN SET$_WRITEERR OR ST$SM_INHIB_MSG;
1754 1746 2 END;
1755 1747 2
1756 1748 2 ! Now that the input and output objects are open, copy the ACL if necessary.
1757 1749 2
1758 1750 2 SACL_CONTEXT = 0;
1759 1751 2
1760 1752 2 WHILE 1
1761 1753 2 DO
1762 1754 2 BEGIN
1763 1755 2 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_READACE;
1764 1756 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = ACL$S_READACE;
1765 1757 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1766 1758 2 STATUS = $CHANGE_ACL (CHAN = .5CHAN,
1767 1759 2 OBJTYP = OBJECT_TYPE,
1768 1760 2 OBJNAM = OBJECT_DESC,
1769 1761 2 ITMLST = ATR_ARGLIST,
1770 1762 2 CONXT = SACL_CONTEXT);
1771 1763 2 IF NOT .STATUS
1772 1764 2 THEN
1773 1765 2 BEGIN
1774 1766 2
1775 1767 2 ! Check for the end of the ACL.
1776 1768 2
1777 1769 2 IF .STATUS EQL SS$_ACLEPTY OR .STATUS EQL SS$_NOMOREACE THEN EXITLOOP;
1778 1770 2
1779 1771 2 ! Not the end, return the error.
1780 1772 2
1781 1773 2 SIGNAL (SET$_READERR, 1, .OBJECT_DESC, .STATUS, 0);
1782 1774 2 RETURN SET$_READERR OR ST$SM_INHIB_MSG;
1783 1775 2 END;
1784 1776 2
1785 1777 2 ! If possible, copy the ACE to the target object.
1786 1778 2
1787 1779 2 IF NOT .ACE[ACE$V_NOPROPAGATE]
1788 1780 2 AND (IF .FLAGS[QUAL_DEFAULT]
1789 1781 2 THEN .ACE[ACE$V_DEFAULT] OR .FLAGS[DIRECTORY]
1790 1782 2 ELSE NOT .ACE[ACE$V_HIDDEN])
1791 1783 2 THEN
1792 1784 2 BEGIN
1793 1785 2
1794 1786 2 ! If this is a default ACE and the target is not a directory file, clear the
1795 1787 2 ! default option in the ACE.
1796 1788 2
1797 1789 2 IF .FLAGS[QUAL_DEFAULT]
1798 1790 2 THEN IF .ACE[ACE$V_DEFAULT]
1799 1791 2 AND NOT .FLAGS[DIRECTORY]
1800 1792 2 THEN ACE[ACE$V_DEFAULT] = 0;
1801 1793 2
1802 1794 2 ! Now add the ACE to the object's ACL.
1803 1795 2
1804 1796 2 ACL_CONTEXT = -1;
1805 1797 2 ATR_ARGLIST[0, ITMSW_ITMCD] = ACL$C_ADDACLENT;
1806 1798 2 ATR_ARGLIST[0, ITMSW_BUFSIZ] = .ACE[ACE$B_SIZE];
1807 1799 2 ATR_ARGLIST[0, ITMSL_BUFADR] = ACE;
1808 1800 2 STATUS = $CHANGE_ACL (CHAN = .CHAN,
1809 1801 2 OBJTYP = OBJECT_TYPE,
```

```
1810 P 1802 4
1811 P 1803 4
1812 1804 4
1813 1805 4
1814 1806 4
1815 1807 4
1816 1808 4
1817 1809 4
1818 1810 4
1819 1811 4
1820 1812 4
1821 1813 4
1822 1814 4
1823 1815 4
1824 1816 4
1825 1817 4
1826 1818 4

OBJNAM = .OBJECT_NAME_DESC,
ITMLST = ATR_ARGLIST,
CONXT = ACL_CONTEXT;

IF NOT .STATUS
THEN
BEGIN
SIGNAL (SET$WRITEERR, 1, .OBJECT_NAME_DESC, .STATUS, 0);
RETURN SET$WRITEERR OR $TSM_INHIB_MSG;
END;
END;

! Now that the ACL has been copied, return to clean things up.
RETURN 1;
END;
```

! End of routine COPY_ACL

```
003C 00000 COPY_ACL:
55 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5
54 00000000G 00 9E 00009 MOVAB LIB$SIGNAL, R5
53 00000000G CF 9E 00010 MOVAB SY$CHANGE_ACL, R4
5E 00000000G AE 9E 00015 MOVAB ATR_ARGLIST, R3
63 000600FF 8F D0 00019 MOVL #393471, ATR_ARGLIST
04 A3 38 ACE, ATR_ARGLIST+4
FCE0 C3 9F 00025 PUSHAB ACL_CONTEXT
7E 7C 00029 CLRQ -(SP)
53 DD 0002B PUSHL R3
04 AC DD 0002D PUSHL OBJECT_NAME_DESC
F9C0 C3 9F 00030 PUSHAB OBJECT_TYPE
FDC C3 DD 00034 PUSHL CHAN
64 07 FB 00038 CALLS #7, SY$CHANGE_ACL
52 50 D0 0003B MOVL R0, STATUS
21 52 E8 0003E BLBS STATUS, 2$
7E D4 00041 CLRL -(SP)
52 DD 00043 PUSHL STATUS
04 AC DD 00045 PUSHL OBJECT_NAME_DESC
01 DD 00048 PUSHL #1
007710D4 8F DD 0004A PUSHL #7803092
65 05 FB 00050 CALLS #5, LIB$SIGNAL
03 00 ED 00053 CMPZV #0, #3, WORST_ERROR, #4
03 19 0005A BLSS 1$
00F4 31 0005C BRW 13$
00E8 31 0005F BRW 12$
1$:
2$:
3$:
04 63 000900FF 8F D0 00065 MOVL #590079, ATR_ARGLIST
A3 38 ACE, ATR_ARGLIST+4
A8 A3 9F 00071 PUSHAB SACL_CONTEXT
7E 7C 00074 CLRQ -(SP)
53 DD 00076 PUSHL R3
FCEC C3 9F 00078 PUSHAB SOBJECT_DESC
FCE8 C3 9F 0007C PUSHAB SOBJECT_TYPE
```

			64	A4	A3	DD	00080	PUSHL	SCHAN		
			52		07	FB	00083	CALLS	#7, SYS\$CHANGE_ACL		
			42		50	D0	00086	MOVL	R0, STATUS		
		000009D0	8F		52	E8	00089	BLBS	STATUS, 7\$	1763	
					52	D1	0008C	CMPL	STATUS, #2512	1769	
		000009E0	8F		07	13	00093	BEQL	4\$		
					52	D1	00095	CMPL	STATUS, #2528		
					03	12	0009C	BNEQ	5\$		
					00BA	31	0009E	BRW	14\$		
					7E	D4	000A1	CLRL	-(SP)	1773	
					52	DD	000A3	PUSHL	STATUS		
				FCEC	C3	DD	000A5	PUSHL	SUBJECT_DESC		
					01	DD	000A9	PUSHL	#1		
				007710B4	8F	DD	000AB	PUSHL	#7803060		
			65		05	FB	000B1	CALLS	#5, LIB\$SIGNAL		
04	F9B8	C3	03		00	ED	000B4	CMPZV	#0, #3, WORST_ERROR, #4		
					09	18	000BB	BGEQ	6\$		
		F9B8	C3	107710B4	8F	D0	000BD	MOVL	#276238516, WORST_ERROR	1774	
			50	107710B4	8F	D0	000C6	MOVL	#276238516, R0		
					04	000CD		RET			
		92	3B	A3	03	E0	000CE	BBS	#3, ACE+3, 3\$	1779	
		0C	F9B4	C3	06	E1	000D3	BBC	#6, FLAGS, 8\$	1780	
				10	A3	E8	000D9	BLBS	ACE+3, 10\$	1781	
		82	F9B5	C3	02	E1	000DD	BBC	#2, FLAGS+1, 3\$		
					08	11	000E3	BRB	10\$		
		03	3B	A3	02	E1	000E5	BBC	#2, ACE+3, 10\$	1782	
					FF78	31	000EA	BRW	3\$		
		0E	F9B4	C3	06	E1	000ED	BBC	#6, FLAGS, 11\$	1789	
				0A	A3	E9	000F3	BLBC	ACE+3, 11\$	1790	
		04	F9B5	C3	02	E0	000F7	BBS	#2, FLAGS+1, 11\$	1791	
			3B	A3	01	8A	000FD	BICB2	#1, ACE+3	1792	
		FCE0	C3		01	CE	00101	MNEGL	#1, ACL_CONTEXT	1796	
			02	A3	01	B0	00106	MOVW	#1, ATR_ARGLIST+2	1797	
			63	38	A3	9B	0010A	MOVZBW	ACE, ATR_ARGLIST	1798	
		04	A3	38	A3	9E	0010E	MOVAB	ACE, ATR_ARGLIST+4	1799	
				FCE0	C3	9F	00113	PUSHAB	ACL_CONTEXT	1804	
					7E	7C	00117	CLRQ	-(SP)		
					53	DD	00119	PUSHL	R3		
				04	AC	DD	0011B	PUSHL	OBJECT_NAME_DESC		
				F9C0	C3	9F	0011E	PUSHAB	OBJECT_TYPE		
				FCDC	C3	DD	00122	PUSHL	CHAN		
			64		07	FB	00126	CALLS	#7, SYS\$CHANGE_ACL		
			52		50	D0	00129	MOVL	R0, STATUS		
			BB		52	E8	0012C	BLBS	STATUS, 9\$	1805	
					7E	D4	0012F	CLRL	-(SP)	1808	
					52	DD	00131	PUSHL	STATUS		
				04	AC	DD	00133	PUSHL	OBJECT_NAME_DESC		
					01	DD	00136	PUSHL	#1		
				007710D4	8F	DD	00138	PUSHL	#7803092		
			65		05	FB	0013E	CALLS	#5, LIB\$SIGNAL		
04	F9B8	C3	03		00	ED	00141	CMPZV	#0, #3, WORST_ERROR, #4		
					09	18	00148	BGEQ	13\$		
		F9B8	C3	107710D4	8F	D0	0014A	MOVL	#276238548, WORST_ERROR	1809	
			50	107710D4	8F	D0	00153	MOVL	#276238548, R0		
					04	0015A		RET			
			50		01	D0	0015B	MOVL	#1, R0	1816	
					04	0015E		RET		1818	

AED\$SETACL
V04-000

J²
16-Sep-1984 00:02:30
14-Sep-1984 11:52:34

VAX-11 Bliss-32 V4.0-742
[ACLEDT.SRC]SETACL.B32;1

Page 68
(9)

; Routine Size: 351 bytes, Routine Base: \$CODES + 1498


```
1828 1819 1 ROUTINE INPUT_ERROR (FILE_FAB) =
1829 1820 1
1830 1821 1 !++
1831 1822 1
1832 1823 1 FUNCTIONAL DESCRIPTION:
1833 1824 1
1834 1825 1 This routine is used to signal errors received on the file scan.
1835 1826 1
1836 1827 1 CALLING SEQUENCE:
1837 1828 1 INPUT_ERROR (ARG1)
1838 1829 1
1839 1830 1 INPUT PARAMETERS:
1840 1831 1 ARG1: address of the FAB
1841 1832 1
1842 1833 1 IMPLICIT INPUTS:
1843 1834 1 none
1844 1835 1
1845 1836 1 OUTPUT PARAMETERS:
1846 1837 1 none
1847 1838 1
1848 1839 1 IMPLICIT OUTPUTS:
1849 1840 1 none
1850 1841 1
1851 1842 1 ROUTINE VALUE:
1852 1843 1 1
1853 1844 1
1854 1845 1 SIDE EFFECTS:
1855 1846 1 The error is signaled by placing the appropriate message into
1856 1847 1 the output file.
1857 1848 1
1858 1849 1 !--
1859 1850 1
1860 1851 2 BEGIN
1861 1852 2
1862 1853 2 MAP
1863 1854 2 FILE_FAB : REF $BLOCK; ! FAB address
1864 1855 2
1865 1856 2 LOCAL
1866 1857 2 STATUS; ! Error to signal;
1867 1858 2
1868 1859 2 STATUS = SET$ OPENOUT;
1869 1860 2 IF .FILE_FAB[FAB$L_STS] EQL RMS$ FNF
1870 1861 2 THEN STATUS = SET$_OPENOUT AND NOT STSM_SEVERITY OR STSK_WARNING;
1871 1862 2
1872 1863 2 FILE_ERROR (.STATUS, .FILE_FAB, .FILE_FAB[FAB$L_STS],
1873 1864 2 .FILE_FAB[FAB$L_STV]);
1874 1865 2
1875 1866 2 RETURN 1;
1876 1867 2
1877 1868 1 END; ! End of routine INPUT_ERROR
```

0000 00000 INPUT_ERROR:
.WORD Save nothing

: 1819

AEDSSETACL
V04-090

L 2
16-Sep-1984 00:02:30 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:52:34 [ACLEDI.SRC]SETACL.B32;1

Page 70
(10)

00018292	51 007710A2	8F D0 00002	MOVL #7803042, STATUS
	50 04	AC D0 00009	MOVL FILE_FAB, R0
	8F 08	A0 D1 0000D	CMPL 8(R0T, #98962
	51 007710A0	07 12 00015	BNEQ 1\$
	7E 08	8F D0 00017	MOVL #7803040, STATUS
		A0 7D 0001E	MOVQ 8(R0), -(SP)
		50 DD 00022	PUSHL R0
0000V	CF	51 DD 00024	PUSHL STATUS
	50	04 FB 00026	CALLS #4, FILE_ERROR
		01 D0 0002B	MOVL #1, R0
		04 0002E	RET

: 1859
: 1860
: 1861
: 1863
: 1866
: 1868

; Routine Size: 47 bytes, Routine Base: \$CODE\$ + 15FA

```
1879 1 ROUTINE FILE_ERROR (ERROR_CODE, FILE_FAB, STS, STV) =
1880 1
1881 1 ++
1882 1
1883 1 FUNCTIONAL DESCRIPTION:
1884 1
1885 1     This routine is used to signal errors received on files.
1886 1
1887 1 CALLING SEQUENCE:
1888 1     FILE_ERROR (ARG1, ARG2, ARG3, ARG4)
1889 1
1890 1 INPUT PARAMETERS:
1891 1     ARG1: error code
1892 1     ARG2: address of the FAB
1893 1     ARG3: primary error status
1894 1     ARG4: secondary error status
1895 1
1896 1 IMPLICIT INPUTS:
1897 1     none
1898 1
1899 1 OUTPUT PARAMETERS:
1900 1     none
1901 1
1902 1 IMPLICIT OUTPUTS:
1903 1     none
1904 1
1905 1 ROUTINE VALUE:
1906 1     1
1907 1
1908 1 SIDE EFFECTS:
1909 1     none
1910 1
1911 1 --
1912 1
1913 2 BEGIN
1914 2
1915 2 MAP
1916 2     FILE_FAB      : REF $BBLOCK;          ! FAB address
1917 2
1918 2 BIND
1919 2     FILE_NAM      = .FILE_FAB[FAB$$_NAM] : $BBLOCK;      ! NAME block address
1920 2
1921 2 LOCAL
1922 2     FILE_NAME      : $BBLOCK [DSC$$_S_BLN];              ! Local file name descr
1923 2
1924 2 CH$FILL (0, DSC$$_S_BLN, FILE_NAME);
1925 2 IF .FILE_NAM[NAM$$_RSL] NEQ 0
1926 2 THEN
1927 2     BEGIN
1928 2         FILE_NAME[DSC$$_LENGTH] = .FILE_NAM[NAM$$_RSL];
1929 2         FILE_NAME[DSC$$_A_POINTER] = .FILE_NAM[NAM$$_RSA];
1930 2     END
1931 2 ELSE IF .FILE_NAM[NAM$$_ESL] NEQ 0
1932 2 THEN
1933 2     BEGIN
1934 2         FILE_NAME[DSC$$_LENGTH] = .FILE_NAM[NAM$$_ESL];
1935 2         FILE_NAME[DSC$$_A_POINTER] = .FILE_NAM[NAM$$_ESA];
```

```
: 1936      1926      3      END
: 1937      1927      3      ELSE
: 1938      1928      3      BEGIN
: 1939      1929      3      FILE_NAME[DSC$W_LENGTH] = .FILE_FAB[FAB$B_FNS];
: 1940      1930      3      FILE_NAME[DSC$A_POINTER] = .FILE_FAB[FAB$C_FNA];
: 1941      1931      3      END;
: 1942      1932      3
: 1943      1933      3      SIGNAL (.ERROR_CODE, 1, FILE_NAME, .STS, .STV);
: 1944      1934      3
: 1945      1935      3      RETURN 1;
: 1946      1936      3
: 1947      1937      1      END;
```

! End of routine FILE_ERROR

				00FC 00000 FILE_ERROR:				
			5E	08	C2 00002	.WORD	Save R2,R3,R4,R5,R6,R7	: 1869
			57	08	AC D0 00005	SUBL2	#8, SP	
			56	28	A7 D0 00009	MOVL	FILE_FAB, R7	: 1909
08		00	6E	00	2C 0000D	MOVL	40(R7), R6	
				6E	00012	MOVC5	#0, (SP), #0, #8, FILE_NAME	: 1914
				03	A6 95 00013	TSTB	3(R6)	: 1915
				08	13 00016	BEQL	1\$	
			6E	03	A6 9B 00018	MOVZBW	3(R6), FILE_NAME	: 1918
	04		AE	04	A6 D0 0001C	MOVL	4(R6), FILE_NAME+4	: 1919
				19	11 00021	BRB	3\$: 1915
				08	A6 95 00023	TSTB	11(R6)	: 1921
				08	13 00026	BEQL	2\$	
			6E	08	A6 9B 00028	MOVZBW	11(R6), FILE_NAME	: 1924
	04		AE	0C	A6 D0 0002C	MOVL	12(R6), FILE_NAME+4	: 1925
				09	11 00031	BRB	3\$: 1921
			6E	34	A7 9B 00033	MOVZBW	52(R7), FILE_NAME	: 1929
	04		AE	2C	A7 D0 00037	MOVL	44(R7), FILE_NAME+4	: 1930
			7E	0C	AC 7D 0003C	MOVQ	STS, -(SP)	: 1933
				08	AE 9F 00040	PUSHAB	FILE_NAME	
				01	DD 00043	PUSHL	#1	
				04	AC DD 00045	PUSHL	ERROR_CODE	
		00000000G	00	05	FB 00048	CALLS	#5, LIB\$SIGNAL	
			1A	04	AC EB 0004F	BLBS	ERROR_CODE, 4\$	
50			03	00	EF 00053	EXTZV	#0, #3, ERROR_CODE, R0	
50	04	AC	03	00	ED 00059	CMPZV	#0, #3, WORST_ERROR, R0	
		CF		08	18 00060	BGEQ	4\$	
				8F	C9 00062	BISL3	#268435456, ERROR_CODE, WORST_ERROR	
	0000'	CF	04	AC	10000000	MOVL	#1, R0	: 1935
			50	01	D0 0006D	RET		: 1937
				04	00070			

; Routine Size: 113 bytes, Routine Base: \$CODE\$ + 1629

```
: 1948      1938      1
: 1949      1939      1      END
: 1950      1940      0      ELUDOM
```


PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	5296	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
LIB\$KEYOS	0	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(1)
LIB\$STATES	14	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(1)
\$PLITS	876	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	5786	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	189	1	1000	00:01.9
\$255\$DUA28:[SYSLIB]TPAMAC.L32;1	42	15	35	14	00:00.2

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SETACL/OBJ=OBJ\$:SETACL MSRC\$:SETACL/UPDATE=(ENH\$:SETACL)

: Size: 5786 code + 6186 data bytes
: Run Time: 01:37.2
: Elapsed Time: 04:37.6
: Lines/CPU Min: 1197
: Lexemes/CPU-Min: 27511
: Memory Used: 578 pages
: Compilation Complete

0004 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

REDMESSAG
LIS

REDPROMPT
LIS

SETACL
LIS

REDSUBR
LIS

0005 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY